# CASH: Classifier of Archetypes and Styles of Humans

Adrian Machado
*Computer Engineering*
*University of Waterloo*
Waterloo, Canada
amachado@uwaterloo.ca

Jenny Dong
*Mechatronics Engineering*
*University of Waterloo*
Waterloo, Canada
jj2dong@uwaterloo.ca

***Abstract*—We introduce CASH, a deep convolutional neural network for human archetype classification based on appearance in photos. This has applications in open casting calls, where agencies receive applications from the public and must manually parse through images to find candidates that have the ”look” or archetype they are casting for. The neural network utilizes MobileNetV2 for image feature extraction, and proceeds to train a fully connected network to classify images in our custom dataset of four different archetypes: business, athletic, fashion, and punk. Diversity was a particular focus in this project as well, with additional training and validation sets consisting of equal racial and gender diversity used to develop the weights, and compare results. CASH was able to achieve a validation accuracy of 75% on a random assortment of headshots, and a 62% accuracy on a diversity-focused validation set.**

***Index Terms*—archetype, classification, neural network, style, machine learning**

## I. Introduction

### A. Background

In the media and advertising industry, there are often casting calls for specific looks in which applicants submit pictures of themselves to be analyzed by the casting agency. Examples of looks include ”hipster”, ”business”, and so on. Many of these looks are actually archetypes or subcultures that extend beyond just the clothing, with athletes being characterized by their physique, or fashion models by their makeup. A common issue in these casting calls is that there tends to be a surplus of random applicants with varying degrees of appropriate looks, which have to be manually pruned. As one can imagine, this can be time-consuming and ineffective process. CASH was developed to replace this manual process by applying classification on a set of applicant images, revealing candidates that match the desired archetype.

Open casting calls normally consist of individuals sending in photos of themselves to agencies, typically headshots. Casting agents then look through the applicants to see who matches their perception of that look. The issue with open casting calls is that there is a huge variety of people who apply, with many matching the desired look to varying degrees based on their facial features and attire. CASH would be useful in scenarios where the agency has a large set of applicant images and which normally need to be manually parsed to find candidates with the desired look or archetype (ex. business look for a bank campaign). CASH would be able to classify all of the images, finding which ones match the desired archetype automatically. Furthermore, another issue in the current solution of manual selection is bias. How well an applicant matches an archetype is highly subjective, and bias within the various casting agents can result in discrimination against candidates based on their skin colour or gender. Through the use of CASH, bias is centralized within the model, and is addressed during the training of CASH, as detailed in our approach.

Given the scope of this research paper, the dataset we created focused on faces as opposed to full-body photographs. As the dataset is custom made as well, there is a variety in image composition such as background colours and noise.

### B. Related work

Papers done in the space that may be relevant include several models used to classify clothing types [1] [2] [3]. These models are designed to recognize not humans, but apparel only (ex. differentiating between t-shirts and dress shirts). In one study a style classification is performed based on the clothing recognized, however the performance of their SVM model was quite poor, with an accuracy of only 41.38% [4]. In that study, upper-body clothing is classified into categories such as bohemian, rock, business, or seasonal; degree of skin exposure; fabric patterns and colours. Apparel classification is useful in identifying what style the person might be portraying, but is only a portion of what truly contributes to the overarching look of an individual. The overlooked sections include makeup, hairstyle, bone structure, facial expressions, and physique, which are potentially captured by CASH, as it operates on mainly headshots. There are also studies that perform facial recognition using deep networks that analyze facial features found in images however they do not focus on style/look classification like CASH does [5], hence the uniqueness of this research.

## II. Approach

### A. Overview

Our problem involves the processing and analysis of a significant number of images in order to perform classification. Certain spatial properties that are characteristic of an archetype (ex. suits on a business archetype) must be understood as high level features of the archetype, similar to the way humans perceive and classify archetypes in real life. As a result, simply flattening the pixels of an image and feeding them as features to a model would not be acceptable as all spatial information is lost. Convolutional Neural Networks (CNN) are commonly used to perform feature extraction on images while preserving spatial information [6]. In theory, the CNN would be able to extract low-level features from the images (ex. edges and curves) and build upon them through successive filters to form high level features (ex. strong jawlines and raised cheekbones in fashion models). There are several challenges introduced through the decision to use a CNN as a feature extractor however.

### B. Initial Challenges

In order to train a CNN, large amounts of images are needed in order to adjust the weights. The issue with this is that our data collection capabilities were limited. We utilized an open-source tool [1] to scrape Google Images in order to find training data, however many of the images collected were unsuitable for various reasons, limiting our pool of data. Figure 1 is an example of an image that was inadmissible due to most of the person's face being hidden. After manually pruning the scraped images, we had approximately 80 images per class for training.

Fig. 1. Images with face obfuscation were deemed unfit for use.

Another issue recognized early on within the scraped data was inherent racial and gender biases. The images in the business archetype class, as seen in Figure 2, were predominantly Caucasian males, with few minorities or women present, contrasted with the editorial class being mostly Caucasian women. This could inadvertently lead to the network being trained to believe that being Caucasian is associated with being part of the business archetype, potentially leading to discrimination against coloured applicants. Both the issue of little data and bias were addressed in the creation of our data set.

### C. Dataset

*1) Creation:* There were a total of 5 distinct datasets used to train, validate, and test CASH. The datasets contained images corresponding to 4 archetypes/looks: punk, business,

[1]Google Images scraper: https://github.com/hardikvasa/google-images-download

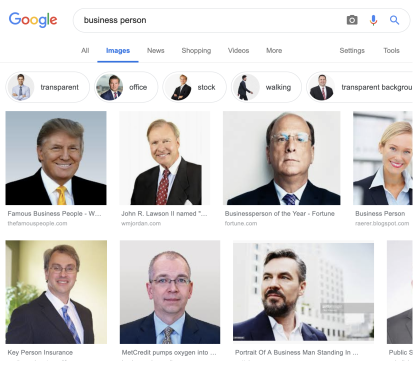

Fig. 2. The majority of business archetype images are Caucasian males by default.

fashion (as in fashion editorial, typical model look), and athletic. These archetypes were chosen as they are commonly found in advertising and media. The images were scraped from Google Images with an open-source API, with varying degrees of manual pruning and diversification being required. The Google Image-specific search query constraints were to have a face type and square image dimensions. The datasets and their characteristics are listed in Table I.

TABLE I
DATA SETS USED

| Dataset | Purpose | Number of Images | Diversity Enforced? |
|---|---|---|---|
| train_generator | Training | 313 | No |
| undiv_validation_generator | Validation | 64 | No |
| div_train_generator | Training | 311 | Yes |
| div_validation_generator | Validation | 130 | Yes |
| test_generator | Testing | 30 | Yes |

The first validation set used was composed of 16 images of each class, with no attention paid to diversity. The diversity-enforced sets were chosen manually through specific Google Image searches (ex. Asian businesswoman vs. business person). An equal distribution across ethnicities (ex. Caucasian, African descent, East Asian descent, South East Asian descent, Latinos) and binary genders was enforced to encourage impartiality within the model. This was only partially achieved for some classes however, due to the lack of certain looks in ethnicities (ex. difficulty finding "punk" for darker-skinned individuals). The test set was similarly developed from selecting a diverse set of images from the diverse validation set, with its small size making it easy to manually analyze where the classifier was making mistakes.

*2) Augmentation:* In order to supplement the lack of training data images, we augmented our existing images in order to create more training samples. The Keras Image Data Generator was used for this, with both a horizontal flip, rescale, and rotation range of up to 5 degrees being used.

```
train_datagen = ImageDataGenerator(
    rescale=1./255,
    horizontal_flip=True,
    rotation_range=5
)
```

Other transformations like vertical flips and more extreme rotations did not make sense for our data, as you would not expect to be given a photograph of someone sideways or upside-down.

Even with these augmentations applied, our data was still limited, and it was unlikely that we would be able to train the weights of a deep CNN. This was addressed by using pre-trained weights however, as discussed in further detail under *Architecture*.

## III. Architecture

### A. Overview

CASH was developed in Google CoLab, featuring Keras and Tensorflow. The network is composed of the MobileNetV2 architecture as the base layers, with custom top layers as show in Table II. Base layer weights were pre-trained on ImageNet. The MobileNets top layers were replaced either with fully connected network (FCN) layers or an SVM layer, depending on the experiment.

TABLE II
CASH Network Layers

| Type/Stride | Filter Shape | Input Size |
|---|---|---|
| Conv / s2 | $3 \times 3 \times 3 \times 32$ | $224 \times 224 \times 3$ |
| Conv dw / s1 | $3 \times 3 \times 32$ dw | $112 \times 112 \times 32$ |
| Conv / s1 | $1 \times 1 \times 32 \times 64$ | $112 \times 112 \times 32$ |
| Conv dw / s2 | $3 \times 3 \times 64$ dw | $112 \times 112 \times 64$ |
| Conv / s1 | $1 \times 1 \times 64 \times 128$ | $56 \times 56 \times 64$ |
| Conv dw / s1 | $3 \times 3 \times 128$ dw | $56 \times 56 \times 128$ |
| Conv / s1 | $1 \times 1 \times 128 \times 128$ | $56 \times 56 \times 128$ |
| Conv dw / s2 | $3 \times 3 \times 128$ dw | $56 \times 56 \times 128$ |
| Conv / s1 | $1 \times 1 \times 128 \times 256$ | $28 \times 28 \times 128$ |
| Conv dw / s1 | $3 \times 3 \times 256$ dw | $28 \times 28 \times 256$ |
| Conv / s1 | $1 \times 1 \times 256 \times 256$ | $28 \times 28 \times 256$ |
| Conv dw / s2 | $3 \times 3 \times 256$ dw | $28 \times 28 \times 256$ |
| Conv / s1 | $1 \times 1 \times 256 \times 512$ | $28 \times 28 \times 256$ |
| $5\times$ Conv dw / s1 | $3 \times 3 \times 512$ dw | $14 \times 14 \times 512$ |
| Conv / s1 | $1 \times 1 \times 512 \times 512$ | $14 \times 14 \times 512$ |
| Conv dw / s2 | $3 \times 3 \times 512$ dw | $14 \times 14 \times 512$ |
| Conv / s1 | $1 \times 1 \times 512 \times 1024$ | $7 \times 7 \times 512$ |
| Conv dw / s2 | $3 \times 3 \times 1024$ dw | $7 \times 7 \times 1024$ |
| Conv / s1 | $1 \times 1 \times 1024 \times 1024$ | $7 \times 7 \times 1024$ |
| Avg Pool / s1 | Pool $7 \times 7$ | $7 \times 7 \times 1024$ |
| FC / s1 | $1024 \times 128$ | $1 \times 1 \times 1024$ |
| FC / s1 | $128 \times 64$ | $1 \times 1 \times 128$ |
| Softmax / s1 | Classifier | $1 \times 1 \times 64$ |

For the initial experiment, the MobileNetV2's top layers were replaced with two fully connected layers with 128 and 64 nodes respectively, and a Softmax layer for classification of the four classes. Due to the vagueness associated with discerning archetypes, multiple fully connected layers are likely necessary as the boundary between classes is likely non-linear and complex. Higher node counts for the FCN were attempted but they did not yield higher accuracy. Dropout is used between the fully connected layers to regularize the network to prevent overfitting from the limited data. L2 regularization is also used on these layers for the same purpose. The second experiment uses SVM for classification instead of Softmax. Further details are provided in the subsequent sections.

### B. Pre-trained Model

Instead of training the weights of the entire CNN ourselves, a pre-trained set of weights was used instead, specifically weights trained on the Imagenet challenge. Images in Imagenet are hierarchical in nature [7], and consist of a wide variety of images, meaning that a pretrained model would likely be able to discern several low-level and high level image features. Styles and archetypes are also hierarchical in natures as well making transfer learning from imagenet weights suitable for our problem [8]. Keras provides several CNN models that are pretrained on Imagenet, with each one having a tradeoff between accuracy and size/training time. We decided to use MobileNetV2 as it provides a large number of parameters while not being too large or slow, as outlined in Figure 3 [9].

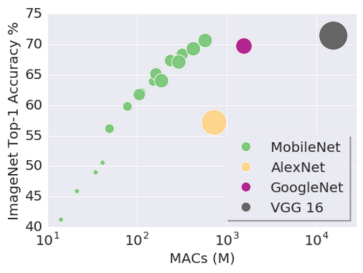

Fig. 3. Mobile Nets provide high imagenet accuracy with a relatively small size [9].

The weights of MobileNetV2s base layers trained on ImageNet were frozen to retain the higher level feature activations [2]. This proved to be particularly beneficial as unfreezing the layers afterwards and retraining yielded poorer results.

### C. Regularization

Another issue that arises from using neural networks on a small data set is overfitting to the training data. Overfitting can result in a model that lacks generalization, or in our case, a model that is unable to handle a wide range of applicant images. Before the introduction of regularization, the model was consistently reaching 99% training accuracy with low validation accuracy, a strong indication of overfitting. The first method used for regularization in the model is dropout on the fully connected layers. By dropping nodes randomly, we teach the nodes on subsequent layers to not overly depends on certain inputs alone [10]. A dropout rate of 0.2 was used on the first FCN layer, and 0.3 was used on the second FCN layer. The second method used was L2 regularization, which enforces simplicity in the weights. The result of this regularization was a drop in training accuracy from 99% to around 91% at the end of training, with an

[2]Keras blog on transfer learning: https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html

increase in validation accuracy by approximately 5%. The complete CASH model with regularization is built as follows:

```
// MobileNetV2 only accept images
// with these dimensions
IMG_SHAPE = (160, 160, 3)

base_model = MobileNetV2(
    input_shape=IMG_SHAPE,
    include_top=False,
    weights='imagenet'
)

// Freezing MobileNet weights
base_model.trainable = False
model = keras.Sequential([
  base_model,
  Flatten(),
  Dense(128, kernel_regularizer=l2(0.01),
  activation='relu'),
  Dropout(0.2),
  Dense(64, kernel_regularizer=l2(0.01),
  activation='relu'),
  Dropout(0.3),
  Dense(num_classes,
  activation='softmax')
])
```

## IV. Experiments

There were a total of four experiments, outlined in Table III. Each of the experiments are detailed in their corresponding sections. The validation accuracy used is the standard Keras `accuracy` metric:

$$accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

For each experiment, training was run for 50 epochs with a batch size of 32 images. Test accuracy was obtained by generating predictions on the test dataset in `test_generator`, and comparing it against the ground truth values.

A mini-experiment was also conducted that dealt with fine-tuning the control and Experiment 2 models in order to potentially obtain a higher accuracy. This is labelled as a mini-experiment as it was a quick change that built upon the other experiments.

In addition to these experiments, there were other minor experiments in parameter tuning such as the FCN node count and the use of additional FCN layers. The performance yielded from this tuning did not result in a significant deviation from the existing results, and thus did not warrant discussion. Table III outlines the major experiments conducted.

### A. Control

The control experiment was performed on the Regular training and validation sets (i.e. they were not pruned for diversity or bias and are simply the acceptable images scraped from the Google Image API query). This was used as the control to demonstrate any variance that occurs from purposeful diversification of the dataset, either in training and/or in validation.

TABLE III
The experiments run with CASH, with various training and validation sets, and model layers

| Experiment | Model | Training set | Validation Set |
|---|---|---|---|
| Control | CASH | Regular | Regular |
| 1 | CASH | Regular | Diverse |
| 2 | CASH | Diverse | Diverse |
| 3 | CASH-SVM | Regular | Regular |

### B. Experiment 1: Regular Training, Diverse Validation

The first experiment performed was to see if CASH performed worse when validated against a validation set that contained near equal representation of different genders and races. Due to the implicit bias found in the normal training data (ex. under-representation of S.East Asians in athlete archetype photos), we hypothesized that the model would be unable to correctly classify diverse applicants. The predicted result is that the accuracy will decrease from the control experiment.

### C. Experiment 2: Diverse Training, Diverse Validation

The second experiment performed was to train and validate the model against a diverse dataset. By exposing the model to a diverse set of applicants in the training data, it was predicted that it would be able to better classify images in the diverse validation set. The reason for this being that the model would ideally learn features that were skin-colour and gender agnostic (ex. suits are a strong indicator of a business archetype). We would expect the accuracy to be higher than that of Experiment 1.

### D. Experiment 3: SVM

A Support Vector Machine (SVM) was used in lieu of the fully connected layers at the output of MobileNetV2. SVMs make use of hyperplanes for class-seperation which are linear, in comparison to fully-connected networks which are non-linear. Additionally, training an SVM is less computationally intensive than training a neural network as there are fewer weights to be adjusted and updated. Studies have also shown that optimization of the margin-based loss compared to the cross-entropy loss used in the FCNs Softmax layer may yield better accuracy [11] as well. We predict that the SVM will perform worse however, as the archetypes do not seem to be linearly separable due to the vagueness in their definition.

### E. Mini-Experiment: Fine-tuning

The control and diverse models underwent further fine-tuning to see if there were any accuracy improvements to be had. This tuning involved unfreezing the MobileNet base layers from the 50th layer onwards, and then retraining for an additional 20 epochs at a lower training rate. We expect training loss to decrease, however this may not result in lower validation loss due to overfitting.

## V. Results

### A. Overview

The results for each of the experiments is detailed in Table IV. Based on these preliminary results, it would appear that training and validation on regular datasets does not offer a huge advantage or disadvantage to the overall accuracy. In Experiment 3, there is even a drop in overall accuracy although one might argue that this is due to the greater range of data variation within the same magnitude of data.

TABLE IV
The experiments run with CASH, with various training and validation sets, and model layers

| Experiment | End Validation Accuracy | Test Accuracy |
|---|---|---|
| Control | 0.75 | 0.63 |
| 1 | 0.62 | 0.66 |
| 2 | 0.5 | 0.5 |
| 3 | 0.26 | 0.3 |

Though the interclass accuracy was not exceedingly high, this is expected given the variety of looks and similarities across the styles. For example, in Figure 4, the punk class was erroneously categorized as fashion. This is unsurprising given similar characteristics and the relatively small dataset that might emphasize these overlaps, such as darker eyeliner and eccentric hairstyles.

### B. Control

The training accuracy, shown in Figure 5, was relatively high – varying between 0.80 - 0.98. Loss was also low, hovering around 0.4 - 1.5, demonstrating overfitting despite the use of regularization. The validation and test set accuracies were quite high considering that the only other study performing style analysis yielded an accuracy of 41% [4]. This lends credence to the idea that a style or archetype is about more than just clothing, as CASH analyzes faces as well. This experiment is used as the baseline for comparison for all other experiments to see if changes in the training set, validation set, or architecture have a significant impact on the models accuracy.

### C. Experiment 1

With the introduction of more diverse samples in the validation set, there is actually a steady downshift in accuracy. This downshift is evident past epoch 15 in Figure 6. The drop is likely due to the new-found variance in applicant appearances in terms of skin colour, hair length and other characteristics. The model cannot rely on certain features it has learned that are tied to over-represented races/genders. The accuracy oscillates around 60%, which is significantly better than random guessing but reflects the ambiguity in distinguishing these classes, especially when race and gender cannot be used. This result is in line with our prediction that accuracy would drop.

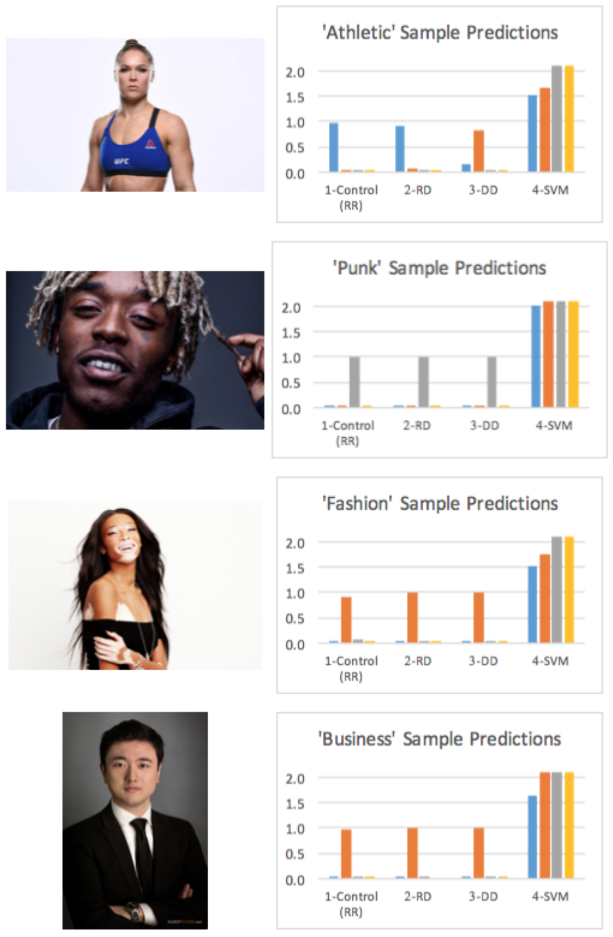

Fig. 4. Histograms for each experimentś predictions on a sample from each class. *Left to right*: Control, Exp. 1, Exp. 2, Exp. 3. *Colours*: Blue-athletic, Orange-business, Grey-fashion, Yellow-punk

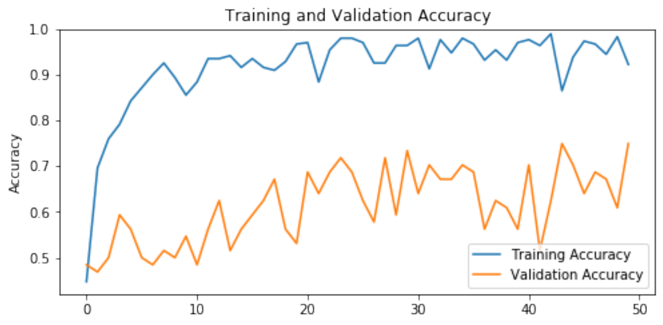

Fig. 5. Training and validation accuracy for the control experiment.

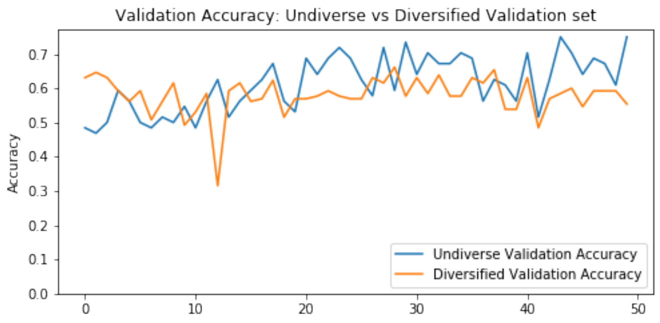

Fig. 6. Validation accuracy of CASH against the Normal and Diversified validation sets.

### D. Experiment 2

When training and validating against diverse sets, there is significantly higher deviation in accuracy compared experiment 1. This is likely due to the hit-or-miss nature of the problem since there are many more high-level features to consider with diversity introduced, such as bone structure or hair. It is also possible that by removing over-represented races/genders, the model was unable to learn their specific features well, leading to originally correct classifications of those samples becoming incorrect in the diversified model. The accuracy, shown in Figure 7, hovers around 50%, which is contrary to our predictions as we expected an increase in accuracy due to the introduction of diversity relative to the Control experiment. This result is still acceptable, given the large increase in variance in the data, and our hypothesis may still hold true for a larger data set.

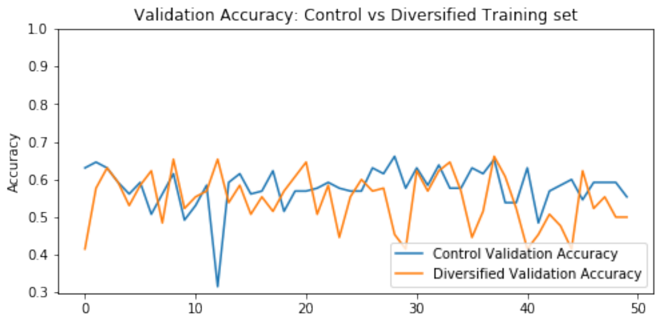

Fig. 7. Validation accuracy of CASH for the control experiment.

### E. Experiment 3

Figure 8 compares the validation accuracy obtained from using an SVM compared to the SoftMax FCN. The results of the SVM model were relatively poor, so despite being faster to train, the algorithm did not offer any real advantages over the other models. The accuracy is less than or half of what FCN CASH is. The low accuracy lends credence to the idea that distinguishing between archetypes is not a simple, linear problem, otherwise the SVM would have similar performance to the FCN.

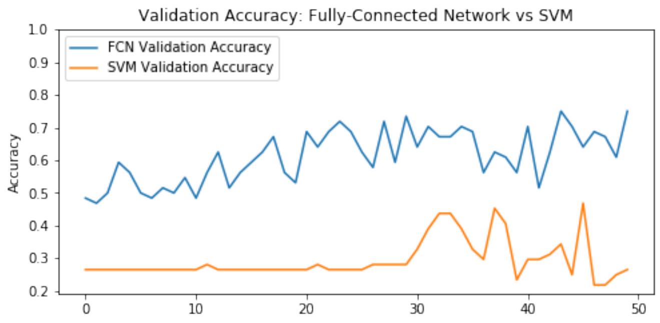

Fig. 8. Validation accuracy of CASH vs CASH-SVM.

### F. Mini-Experiment

Fine-tuning did not offer any sort of improvement to the models accuracy. This is most likely due to the size of the training set being insufficient to properly train the last 50 layers of MobileNetV2 without overfitting.

## VI. Insights

Based on the results from the experiments conducted, several insights can be drawn.

### A. Bias Against Diversity

The initial Google scrapes for the regular data sets exhibited obvious bias. For example, the regular athletic dataset was majority white or black men. The fashion dataset was largely white women. Punk was majority white individuals. The business class had relatively the largest amount of diversity, but was still dominated by males.

The diversity sets were essentially the regular sets but with more samples of diverse data. For example, there are greater numbers of darker-skinned individuals in punk; there are more women in business. The results of this however were negligible in comparison to the solely-regular set pieces, although this should be taken with a grain of salt given the size of the dataset.

### B. Overlaps in Looks

One thing to make note of is the ambiguity and subjectiveness of looks. There are looks that can easily be categorized as one class or the other, which is particularly evident in this paper between fashion and punk. There is actually a punk-fashion trend [3] that exemplifies this overlap, as displayed in Figure 9. To this regard it might be worthwhile to explore the fuzziness of these looks.

This is exaggerated in the poor SVM classification; defining a linear separator between classes is in this case, arguably impossible. Within our dataset there are also figures that are ambiguous, such as Figure 10.

This image is to a human at first glance, an athlete likely due to muscle tone and Nike shirt. Notice, however, the stance of the individual: crossed arms and straight-on smile are

[3]Modern Punk Fashion: https://www.elle.com/uk/fashion/trends/articles/a42203/the-new-punk/

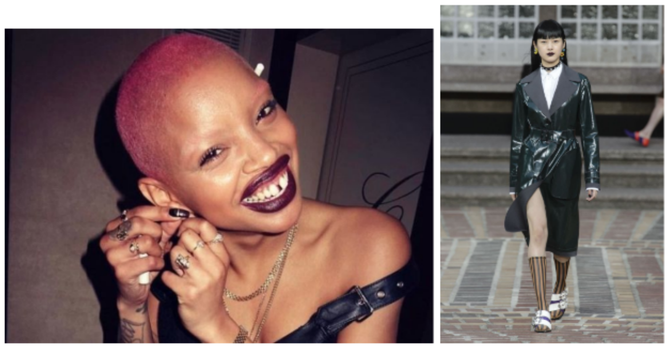

Fig. 9. Punk fashion, which might be classified as both ”punk” and ”fashion”.

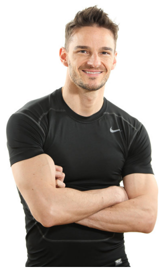

Fig. 10. A test set image classified as both business and as athletic across experiments.

very characteristic of typical business portfolio shots. Black clothing is also often indicative of a business look, which can lead to CASH being fooled.

## VII. Conclusion

When looking for individuals that exhibit a look to fulfill a certain campaign or commercial role, agencies can often be overwhelmed with irrelevant applications during an open casting call. To prune this subset of images, CASH performs decently at identifying different looks between four classes business, fashion, punk, and athletic given a relatively small dataset with accuracy of around 60-70%. This number begins to drop about 10% with purposeful diversity, but this may be caused by insufficient data given the increased feature set within the same dataset size. The relatively minimal impact may also be due to the model learning characteristics beyond gender or ethnicity, and looking instead to features such as eyeliner or attire. Using SVM was enormously unsuccessful, likely due to the ambiguous nature of looks. The suggestions for next steps might be to expand the diversity dataset in response to the wider feature set; to expand diversity to encompass gender-fluidity and interracial individuals; or to use a fuzzier approach to the problem. Given the scope of this paper, however, CASH performs relatively well in identifying four different looks even with diverse sets, and much better than comparable studies.

## References

[1] K. Rasul, H. Xiao, R. Vollgraff, “Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms ,” arXiv e-prints, August 2017.

[2] B. Lao, K. Jagadeesh, “Convolutional Neural Networks for Fashion Classification and Object Detection,” arXiv e-prints, 2015.

[3] H. Chen, A. Gallagher, B. Girod, “Describing Clothing by Semantic Attributes,” ECCV, 2012.

[4] L. Bossard, M. Dantone, C. Leistner, C. Wengert, T. Quack, L. Van Gool, “Apparel Classification with Style,” ACCV, 2012.

[5] M. Wang, W. Deng, “Deep Face Recognition: A Survey,” arXiv e-prints, 2019.

[6] A. Krizhevsky, I. Sutskever, G. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” NIPS, 2012.

[7] J. Deng, W. Dong, R. Socher, L. Li, K. Li, L. Fei-Fei, “ImageNet: A Large-Scale Hierarchical Image Database,” CVPR, 2009.

[8] L. Torrey, J. Shavlik, “Transfer Learning ,” IGI Global, 2010.

[9] A. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” arXiv e-prints, 2017.

[10] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov “Transfer Learning ,” Journal of Machine Learning Research, 2014.

[11] Y. Tang, “Deep Learning using Linear Support Vector Machines ,” arXiv e-prints, 2015.