Predicting Facial Attributes in Video using Temporal Coherence and Motion-Attention

Emily M. Hand Carlos D. Castillo Rama Chellappa
University of Maryland
{emhand,carlos,rama}@umiacs.umd.edu

Abstract

Recent research progress in facial attribute recognition has been dominated by small improvements on the only large-scale publicly available benchmark dataset, CelebA [18]. We propose to extend attribute prediction research to unconstrained videos. Applying attribute models trained on CelebA – a still image dataset – to video data highlights several major problems with current models, including the lack of consideration for both time and motion. Many facial attributes (e.g. gender, hair color) should be consistent throughout a video, however, current models do not produce consistent results. We introduce two methods to increase the consistency and accuracy of attribute responses in videos: a temporal coherence constraint, and a motion-attention mechanism. Both methods work on weakly labeled data, requiring attribute labels for only one frame in a sequence, which we call the anchor frame. The temporal coherence constraint moves the network responses of non-anchor frames toward the responses of anchor frames for each sequence, resulting in more stable and accurate attribute predictions. We use the motion between anchor and non-anchor video frames as an attention mechanism, discarding the information from parts of the non-anchor frame where no motion occurred. This motion-attention focuses the network on the moving parts of the non-anchor frames (i.e. the face). Since there is no large-scale video dataset labeled with attributes, it is essential for attribute models to be able to learn from weakly labeled data. We demonstrate the effectiveness of the proposed methods by evaluating them on the challenging YouTube Faces video dataset [31]. The proposed motion-attention and temporal coherence methods outperform attribute models trained on CelebA, as well as those fine-tuned on video data. To the best of our knowledge, this paper is the first to address the problem of facial attribute prediction in video.

1. Introduction

Facial attributes are high-level, describable visual features of faces that have been used for image search, retrieval, identification and verification [14][15][16]. The problem of accurately recognizing facial attributes in unconstrained environments is very difficult and has gained attention in recent years, with most methods taking advantage of deep learning for feature extraction [18][29][26].

Most facial attributes – *hair color*, *eye color*, *gender* – are invariant to changes in pose, illumination, and various imaging conditions (e.g. resolution). However, without data representing the full spectrum of these different variables, it is difficult to properly model the attributes. CelebA is a large-scale dataset, with over 200,000 images labeled with 40 binary facial attributes [18]. However, when training deep networks, 200,000 images is considered small. And, as the name would suggest, CelebA consists of posed images of celebrities, so the images are of high quality, with good lighting, no motion blur, and the faces are mostly frontal. Due to these biases, and CelebA being the only large-scale publicly available dataset labeled with facial attributes, advances in facial attribute prediction research have been limited to small improvements on this dataset. We propose to shift the focus of facial attribute prediction research to video, which has not yet been explored.

Video data poses many problems for recognition systems, especially those that have only learned from still images. Testing a still image-trained model on data that has a temporal component, in addition to extreme poses, and motion blur, can result in unexpected behavior. In the case of facial attributes, we see the model's response for *gender* changing between video frames, which is not something we expect from a robust attribute classifier. Using what we intuitively know about the stability of attributes over time, we can leverage weakly labeled data to learn more robust attribute models. Most facial attributes are stable over time (e.g. *gender*, *hair color*, *eyeglasses*, etc.), but there are a few that can change throughout a video. In labeling four frames from every sequence in YouTube Faces, we discovered that

there are eight attributes out of the set of forty from CelebA that can change throughout a video: *arched eyebrows*, *bags under eyes*, *blurry*, *double chin*, *hat*, *mouth slightly open*, *narroweyes*, and *smiling*. With 20% of facial attributes being variable over time, we cannot make the assumption that all frames have the same attribute responses.

To better model attributes in video, we propose two methods which use weakly labeled data during training to improve the reliability and accuracy of attribute predictions. Both methods require that one frame in a sequence is labeled with attributes. We call this the anchor frame and all unlabeled frames, non-anchor frames.

We introduce a novel temporal coherence (TC) constraint, which encourages non-anchor video frames to have similar network responses to their corresponding anchor frame. The effect of the temporal coherence constraint weakens as the time increases between the anchor and non-anchor frames. This allows the network to utilize weakly labeled data in order to move attribute responses closer together without enforcing that they are the same between frames.

In addition, we propose a novel motion-attention mechanism for attributes in video. The motion, blur, and pose changes associated with video data tend to cause problems for attribute prediction models (or any recognition system). If we can explicitly account for motion in the learning of the attribute model, it will be less sensitive to such changes. The proposed motion-attention mechanism suppresses input from regions in the non-anchor video frame which have not moved from the anchor frame, effectively focusing the network on regions with motion.

We demonstrate the effectiveness of our methods by testing them on the challenging YouTube Faces dataset [31]. We labeled four frames in every video with forty attributes from CelebA, which we will make publicly available for future research. The first frame in each video is an anchor frame, and is used for training. At test time, we evaluate our methods on the four labeled frames in each video in order to measure stability. We compare the proposed methods with a model trained only on CelebA, as well as a model fine-tuned on the anchor frames from YouTube Faces. Both of the proposed methods outperform these baseline methods, and combining them results in even further improvements. These results demonstrate the need for explicitly accounting for time and motion when training attribute models for use on video data. Though we test the proposed methods on the problem of facial attribute prediction, we note that the methods are very general, and can be applied to any problem which uses a deep network for feature learning and classification of videos.

Contributions: To the best of our knowledge, this paper is the first to study the effects of motion and time on facial attribute prediction, introducing two techniques – temporal coherence and motion-attention – to explicitly account for these variables during training. We also introduce a Multi-Task Attribute CNN (MACNN) as our attribute model, which has fewer than 3 million parameters, trains directly from CelebA in less than an hour on a single GPU, and achieves the same performance as state-of-the-art, [26], on CelebA.

The remainder of the paper is organized as follows: Section 2 discusses previous work related to facial attributes as well as the use of time and motion in video processing. Section 3 details the proposed methods, and Section 4 highlights the experiments performed to test these methods. In section 5 we summarize our work and discuss its impact.

2. Related Work

This paper builds on two areas of research: facial attribute prediction, and recognition from video. We review the relevant literature in the following sections.

2.1. Facial Attributes

Facial attributes – *gender*, *facial hair*, *hair color*, etc – are human describable features of faces and have been successfully used in face recognition and verification [15][16]. The problems of recognizing *gender* and *age* from face images have been studied for many years, and are still considered difficult problems [6][20]. It is well known that the accuracy of *gender* recognition systems degrades significantly in non-ideal settings, i.e. with extreme pose, illumination variations, or poor image quality [28]. Although this work focuses on facial attribute prediction, attributes have been used in many different computer vision problems including describing objects, recognizing activities from video, and face identification and verification [2][3][5][17][10][12][35].

In [14], binary facial attributes were introduced for the task of image search and retrieval. The authors later extended this work in [15] and [16], collecting additional attribute labels and applying attributes to the problem of face verification. As for most applications in computer vision, CNNs have found success in the problem of attribute prediction, with state-of-the-art methods all using CNNs for feature extraction. Pose Aligned Networks for Deep Attribute Modeling (PANDA) was developed for person attribute prediction – *wearing pants*, *long hair*, *male*, etc. – and used part-based CNNs to learn features for each part in a particular pose [32]. Both [25] and [33] utilize a subset of facial attributes in a multi-task learning framework for facial landmark localization. Using an RBM for multi-task attribute learning utilizing facial landmarks for training, [4] achieved state-of-the-art results on facial attribute classification.

Since the introduction of a large-scale attribute-labeled dataset, CelebA, and the addition of attribute labels to LFW,

many new methods have been introduced using deep CNNs for feature extraction [18]. In [18], the authors frame the problem of attribute recognition from faces as one of face localization and attribute prediction. They use two deep networks: LNet and ANet, the first for localizing faces with weak attribute supervision, and the second for attribute prediction from the localized face. Using wearable camera data for pre-training, [29] outperformed [18] on CelebA. The authors collected face tracks from wearable cameras and pretrained their network on the task of verification before finetuning for attribute prediction. Taking advantage of the relationships amongst attributes, [9] is able to improve prediction accuracy and train directly from CelebA. In [26], the authors introduced a mixed objective optimization network for attribute prediction, surpassing state-of-the-art results on CelebA. Their network employed a domain-adaptive reweighting of the error back-propagation to correct for the imbalance in attribute labels in the training data. [8] uses a technique called Selective Learning to model a specified target distribution for each attribute, accounting for the label imbalance problem in CelebA and achieveing the current state-of-the-art in attribute recognition. With CelebA as the only large-scale attribute dataset available for training, progress on attribute prediction has slowly inched forward on the benchmark over the last few years. Rather than aiming to improve prediction accuracy on this benchmark, we instead focus on constructing more robust attribute models by explicitly accounting for temporal variations and motion.

2.2. Recognition from Video

There have been several decades of research in automated video processing [1][22][24][34]. Here we review some recent publications which are related to our work.

There are several datasets labeled with attributes for actions, however the attributes are labeled for each action, not for each video, and certainly not for each frame. For example, for the action *applying lipstick*, there is an attribute *arm up*. If in some frame the subject's arm is not up, that frame is still labeled as such [21][27].

The concept of temporal coherence and feature stability in video has been applied to deep networks in the past: In [11], the authors introduced the concept of steady feature analysis, which aims to learn invariant features from unlabeled data for use in recognition tasks. Rather than encouraging features between video frames to be similar, steady feature analysis encourages feature changes to be smooth, placing constraints on the higher order derivatives of the feature space. Altering stochastic gradient descent to apply a coherence constraint to unlabeled video frames while at the same time learning from images with labels, [19] learned models for different recognition problems by leveraging labeled and unlabeled information. To perform unsupervised feature learning using autoencoders, [7] used temporal coherence to leverage unlabeled data. In [30], they learned feature representations with unlabeled video using tracking as the only supervision. They enforced triplet constraints at every frame according to the query, the tracked object and a random patch from the frame that does not overlap with the tracked object. Our temporal coherence constraint differs from past work as it takes advantage of the relationship between non-consecutive frames, with frames closer together having a larger effect on learning than frames that are farther away from each other. The proposed temporal coherence constraint also differs from previous work in that it operates in a semi-supervised setting, requiring labels only for one frame in a video.

Layer	Parameters/Activation/Pooling/Norm
Conv1	100 5x5 Filters ReLU Max Pooling 3x3 LRN 5x5
Conv2	200 3x3 Filters ReLU Max Pooling 3x3 LRN 5x5
Conv3	300 3x3 Filters ReLU Max Pooling 5x5 LRN 5x5
Conv4	300 5x5 Filters ReLU
FC1	40 Units

Table 1: MACNN Architecture. Conv1 is the bottom layer, and FC1 is the top and final layer producing 40 outputs.

In [23], the authors used the magnitude of the optical flow to amplify features for action recognition. The idea here is that parts of the image with more motion than others will correspond to higher weighted features for action recognition. We use motion differently in our work, as an attention mechanism rather than a feature weighting technique. The proposed motion-attention focuses the deep network on areas of the video where motion occurs (i.e. the face), removing information from the background. Motion-attention works as a kind of regularizer, keeping the network from overfitting during the fine-tuning process, as we discuss later.

3. Proposed Method

3.1. Multi-Task Attribute CNN

We use a multi-task attribute CNN (MACNN) for feature learning and classification. All attributes are learned simultaneously in MACNN. MACNN's architecture is detailed in table 1. MACNN has a very small architecture, with only four convolution layers, and one fully connected layer added for classification. The network has fewer than

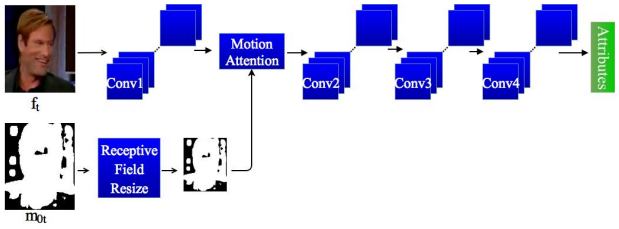

Figure 1: A visualization of the proposed motion-attention technique.

3 million parameters. AttCNN with Selective Learning, the state-of-the-art method for attribute prediction, has roughly 6 million parameters [8]. MACNN has fewer than half of the parameters of AttCNN. The proposed Motion-Attention and Temporal Coherence methods discussed in the following sections use MACNN as the base network.

3.2. Motion-Attention

We introduce a novel attention technique based on motion in a video. Given two consecutive (or nearly consecutive) frames of a face video, we expect the attributes to remain the same, even if there is a small amount of motion. This is a reasonable assumption as most videos are captured at more than 20 frames per second and the face should not change much in $\frac{1}{20}$ of a second. In order to account for motion in a video, we introduce an attention mechanism based on motion between frames. This motion-attention mechanism is applied in a CNN, focusing the network on regions of motion, suppressing input from regions of a video frame where no motion occurred. The intuition here is that between nearly consecutive frames in a video taken from a stationary camera, the motion will occur only in the portion of the frames containing the face. Therefore, motion-attention will focus the network on the face, suppressing information from the background.

For each pair of consecutive frames (f_i, f_j), we have an associated binary optical flow image m_{ij}, which is the result of thresholding the optical flow between f_i and f_j. That is, m_{ij} has a value of 1 where there is motion between f_i and f_j, and a 0 where there is no motion. Optical flow images can be combined for non-consecutive frames by taking the maximum value at each location for all flow images. For example, if we have three consecutive frames, f_i, f_j, and f_k, we can compute m_{ik} by taking the maximum of both m_{ij} and m_{jk} for every location in the image. That is, for all (x, y), $m_{ik}(x, y) = \max(m_{ij}(x, y), m_{jk}(x, y))$. So, the binary flow image for non-consecutive frames has a 1 at every location where there is motion between any of the frames, and a 0 where there is no motion between any of the frames. These binary flow images are used as attention maps to focus the network on regions of motion.

For each video, we have one anchor frame f_0, and the rest are non-anchor frames f_t, $t \geq 1$. The anchor frame has attribute labels, and the non-anchor frames are unlabeled. For each pair of frames (f_0, f_t) we have the corresponding binary motion frame m_{0t}. The process of computing m_{0t} is described above. Each m_{0t} is used to turn off activations for parts of f_t where there is no motion, acting as an attention mechanism for the network. This encourages the network to focus on the parts of f_t which moved (i.e. the face), suppressing the portions of f_t which did not move (i.e. background). We refer to this method as a motion-attention mechanism. Figure 1 visualizes the proposed motion-attention mechanism, with two paths, one for the frame f_t, and one for the binary flow image m_{0t}.

Note that in figure 1, the motion-attention mechanism is being applied after the first convolution layer, but the motion-attention can be applied at any point in the network. Since motion-attention can be applied at any point in the network, m_{0t} must first be resized to match the size of the feature maps where it will be applied. That is, if a layer L produces n_L feature maps of size w_L by h_L, then m_{0t} must be resized to w_L by h_L and then multiplied element-wise with all n_L feature maps, producing n_L focused feature maps. A receptive-field resizing is applied to m_{0t} so that each neuron in the resized m_{0t} has the same receptive field as the neuron in L where it is applied. That is if a neuron in the resized m_{0t} has a value of 1, then it means that there was motion between f_0 and f_t in the receptive field of that neuron, and similarly if it has a value of 0, then there was no motion between f_0 and f_t in the neuron's receptive field.

The motion-attention mechanism is applied during training. In the forward pass, m_{0t} is used to turn off the activation for neurons corresponding to regions without motion, and in the backward pass, no learning is performed for those neurons. We use both anchor and non-anchor frames as input to the network. For the anchor frames, f_0, we define m_{00} to be an image with all 1s, so no attention is used on the anchor frames. As noted earlier, the non-anchor frames f_t do not have attribute labels, and so the labels for the anchor frames are used as the labels for the non-anchor frames. The motion-attention works as a type of regularization deterring the network from over-fitting to the training data. By focusing the network on regions of motion in the non-anchor frame, motion-attention forces the network to explicitly account for motion and to learn how and what motion affects the labels.

Consider a video of a person talking. Let's say the anchor frame f_0 is labeled as a positive instance of *mouth slightly open*. We can assume that in f_1 the person's mouth will still be open, but perhaps a little more or less open, depending on what they are saying. In this instance, m_{01} may be 0 everywhere but around the mouth, where it is 1. So the motion-attention mechanism would focus the network

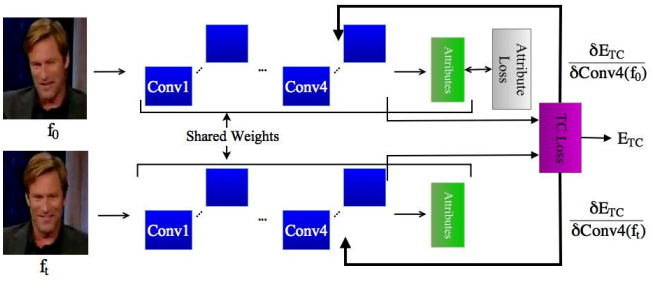

Figure 2: Visualization of two streams using the temporal coherence loss.

on the mouth. Since the subject's mouth is likely still open, we use the label from f_0 for f_1, and now the network is only looking at the mouth, allowing the network to learn a more robust representation for *mouth slightly open*. The proposed motion-attention mechanism can be used alone, or with the proposed temporal coherence constraint described below.

3.3. Temporal Coherence

We introduce a Temporal Coherence (TC) constraint which works in a multi-stream network. A single-stream network is a normal network which takes an input image (or images), and outputs a label (or set of labels). A multi-stream network contains multiple copies of a single stream network, all sharing the same weights, each with their own input and output, and the streams are connected in some way. The TC constraint is implemented as a loss that takes two layers as input, one from the first stream and the corresponding layer from another stream. The first stream of the network acts on an anchor frame of a video (f_0), and the other streams act on non-anchor frames some time t away from f_0 (f_t). Specifically, the second stream has f_1 as input, the third stream has f_2 as input and so on. The weights are shared among all the streams. The TC loss can be attached between any two streams at any point in the network, and is visualized in figure 2 with two streams.

Let F_i be the single stream CNN associated with input frame f_i. Since all streams in the network share the same weights, instead of referring to each stream individually, we will simplify notation by ignoring the stream, as the input indicates the stream. Let $F^l(f_0)$ and $F^l(f_t)$ be the l^{th} layer's activations for f_0 and f_t input respectively. The TC loss aims to move $F^l(f_t)$ toward $F^l(f_0)$, taking into account the distance t. That is, for a robust attribute model, we expect the activations for the anchor frame (f_0) to be similar to the activations for a non-anchor frame (f_t), assuming that not too much time has passed between f_0 and f_t. We also expect that the activations for frames that are closer together will be more similar than the activations for frames that are farther away.

Since f_0 is an anchor frame, we have labels for f_0, and so we apply a multi-label attribute loss on the first stream of the network (F_0), which takes f_0 as input. For F_t, the stream with the non-anchor frame f_t as input, we apply the TC loss in order to move the activations of F_t towards those of F_0. In other words, the error from the TC loss is only propagated through the F_t stream, not the F_0 stream, and the multi-label loss error is only propagated through the F_0 stream since we only have labels for f_0.

More formally, the TC error for layer l is given in equation (1), where $\lambda(t)$ is some non-increasing positive function of t. $\lambda(t)$ is essentially the effect of the frame difference on the error, and therefore on learning. It makes sense for $\lambda(t)$ to not increase as t increases, as the farther two frames are from each other in a video sequence, the less likely they are to be similar, and so the effect of the error between their activations should be less. Equation (1) is used in the forward pass of the network using a TC loss, and equations (2) and (3) are used in the backward pass, propagating the error back through the network. Equation (2) indicates that the error only back-propagates for non-anchor frame inputs. A visualization of the proposed TC loss is shown in figure 2.

$$E_{TC} = \frac{\lambda(t)}{2}||F^l(f_t) - F^l(f_0)||_2^2 \tag{1}$$

$$\frac{\delta E_{TC}}{\delta F^l(f_0)} = 0 \tag{2}$$

$$\frac{\delta E_{TC}}{\delta F^l(f_t)} = \lambda(t)(F^l(f_t) - F^l(f_0)) \tag{3}$$

One may ask why we chose this formulation rather than assuming f_t has the same attribute labels as f_0. We illustrate this with an example: Let's say that we have two consecutive frames, f_0 and f_1. In f_0, the subject is frontal, there is no blur, and the subject does not have *arched eyebrows* (i.e. *arched eyebrows* is labeled as negative). In f_1, the subject moves, raising their eyebrows. The label for *arched eyebrows* in f_0 no longer applies. Our goal is to make the attribute model more robust using video data, and so we do not want the network to make a decision based on something it does not know. Therefore, we want to move the activations for f_1 towards not *arched eyebrows*, but not necessarily label f_1 as a negative instance of *arched eyebrows*. This gives us some intuition as to why we do not want to label f_1, or any f_t, with the same attributes as f_0. As we will see in our experiments, the proposed TC constraint improves attribute predictions over fine-tuning with f_t labeled with the attributes from f_0. The proposed Temporal Coherence constraint is able to utilize weakly labeled data to make a more robust attribute model without having access to labels for all frames.

In labeling four frames from every video in YouTube-Faces we found that there are eight attributes from the forty labeled in CelebA that can vary throughout a video: *arched eyebrows*, *bags under eyes*, *blurry*, *double chin*, *hat*, *mouth*

(a) Arched Eyebrows (b) Bags Under Eyes (c) Blurry (d) Double Chin (e) Hat (f) Mouth Slightly Open (g) Narrow Eyes (h) Smiling

Figure 3: Samples from YouTubeFaces where attributes change between frames. In (a), the top frame shows the man having *arched eyebrows*, but in the bottom frame he does not. (b) shows a frame where the woman has *bags under eyes* and then a frame where she does not. Similarly for (c)-(h)

slightly open, *narroweyes*, and *smiling*. Figure 3 shows some examples of videos where these attributes change.

4. Experiments

4.1. Data

4.1.1 CelebA

CelebA is an attribute-labeled dataset of face images [18]. The dataset has roughly $200,000$ images, with $160,000$ for training, and $20,000$ each for validation and testing. Each image in the dataset is labeled with 40 binary facial attributes including *male*, *blond hair*, *heavy makeup*, etc. CelebA contains mostly frontal, posed images of celebrities. We use the aligned 178x218 CelebA images for training our base attribute model, MACNN.

4.1.2 YouTube Faces

The video dataset in this work is YouTube Faces. YouTube Faces is a video face verification dataset consisting of $3,425$ videos of celebrities from YouTube, with a total of roughly $620,000$ frames [31]. The data varies significantly from CelebA in quality, resolution, lighting, and pose. We labeled four frames in every video with the 40 binary attributes from CelebA. The four frames correspond to the first frame, one from a third of the way through the video, one from two-thirds of the way through the video and the last frame: T_0, T_1, T_2, and T_3 respectively. For benchmarking purposes, there are 10 splits provided with the data for cross-validation testing. We use the anchor (attribute-labeled first frame) and non-anchor (no attribute labels) frames from the training portion of each split to fine-tune MACNN with our different methods. In all our experiments, and we test on the labeled frames from the test splits, and average over all 10 splits. We use the face boxes provided with the dataset, extracting each face from its original frame, and resizing it to 178×218, the same size as the aligned CelebA images. We do not perform alignment on the YouTube Faces frames, as attributes should be invariant to such mis-alignments.

Method	Average Accuracy
LNet+ANet [18]	87.3%
Walk and Learn [29]	88.1%
MOON [26]	**90.90%**
AttCNN [8]	**91.05%**
MACNN (Ours)	**90.9%**

Table 2: Average attribute accuracy on the CelebA test set.

4.2. MACNN

We implement and test MACNN using Caffe [13]. MACNN is trained from scratch using only the aligned CelebA training images, without pre-training on an external dataset. A sigmoid cross-entropy loss is used to facilitate training with batches of size 100. As pre-processing steps, we subtract the training mean from all images and take random crops of 178x178 from each 178x218 input image. After 53 epochs, the error on the validation set no longer decreases, so training is stopped. We note that since MACNN only has 3 million parameters, it takes less than an hour to train on a single GPU.

We compare MACNN with state-of-the-art methods for attribute prediction in Table 2 to show that it is a good starting point for our temporal coherence and motion-attention work. From Table 2, we see that MACNN performs roughly as well as AttCNN – the current state-of-the-art – on average.

We use MACNN for our experiments rather than AttCNN because AttCNN has roughly 6 million parameters, while MACNN has only 3 million parameters, is very quick to train, and is less likely to over-fit. Testing MACNN on the anchor frames of YouTube Faces resulted in an average attribute accuracy of **83.80%**. The average attribute accuracy on YouTube Faces anchor frames is computed as follows. For each split, we compute the accuracy for each at-

Model	T_0	T_1	T_2	T_3	Average
$MACNN_0$	86.55	86.57	86.44	86.23	86.44
$MACNN_1$	86.67	86.64	86.58	86.40	**86.57**
$MACNN_2$	86.55	86.61	86.56	86.20	86.48
$MACNN_3$	86.41	86.46	86.42	86.12	86.35
$MACNN_{10}$	85.68	85.75	85.58	85.53	85.64

Table 3: Average attribute accuracy on YouTube Faces labeled test (T_0, T_1, T_2, T_3) frames fine-tuning with anchor and non-anchor frames using the anchor labels.

tribute, giving us 40 attribute accuracies for each split. Averaging the accuracies for all 40 attributes gives us a single average attribute accuracy for each split. We then average this average attribute accuracy for the 10 splits, giving us **83.80%** for MACNN. For all of the experiments below, we start with MACNN (trained on CelebA), and we fine-tune it on YouTube Faces using four different methods: fine-tuning with anchor and non-anchor frames, using anchor labels for both (4.3), using motion-attention (4.4), using temporal coherence (4.5), and using motion-attention and temporal coherence (4.6).

4.3. Fine-Tuning

Taking a model trained on still images, and fine-tuning it on labeled video data is one way to adjust the model to better handle video data. We do that here by fine-tuning MACNN on the anchor frames for each split in YouTube Faces using a sigmoid cross-entropy loss on the attributes. We call this fine-tuned network $MACNN_0$, which we evaluate on the test portion for that split. We also fine-tune MACNN on non-anchor frames as well as the anchor frames. Non-anchor frames do not have labels, so we assume that the non-anchor frame has the same labels as its corresponding anchor frame. $MACNN_1$ is MACNN fine-tuned on f_0 and f_1 with both using the labels from f_0, $MACNN_2$ is MACNN fine-tuned on f_0, f_1, and f_2 all with the labels from f_0, and so on. Remember that f_0 is the anchor frame and f_1, f_2, etc. are non-anchor frames.

We evaluate our $MACNN_i$ on the labeled test data for each split (T_0, T_1, T_2, T_3) providing the accuracies over all splits on each of the four frames as a measure of stability. Table 3 shows the average attribute accuracies, averaged over the 10 splits for $MACNN_i$ for $i = 0, 1, 2, 3, 10$ on each of the four labeled frames per sequence. Fine-tuning on the anchor frames, $MACNN_0$, provides a 2.5% improvement over the original MACNN, which had an average accuracy of 83.80%. However, the improvements do not continue as more non-anchor frames are added to the training set for fine-tuning. In fact, $MACNN_3$ produces worse results than fine-tuning on the anchor frames alone. $MACNN_3$ is fine-tuning with f_0 and 3 non-anchor frames using the same labels as f_0. We expected the performance of fine-tuning using anchor labels on non-anchor frames would degrade as more non-anchor frames were added because as non-anchor frames move farther away from the anchor frames, it becomes less likely that they share the same label. We also note the sharp decline in performance between $MACNN_3$ and $MACNN_{10}$. We believe this to be due to two factors: one being that simply fine-tuning using anchor labels for non-anchor frames leads to overfitting of the network, and the other being that the assumption that f_0 and f_t have the same, or similar, attribute responses breaks down as t gets larger. We see a similar, though less severe, phenomenon with TC_i and $MATC_i$ in the following sections.

Model	T_0	T_1	T_2	T_3	Average
MA_1	86.73	86.82	86.78	86.50	86.70
MA_2	86.91	86.97	86.94	85.65	86.86
MA_3	86.95	87.09	86.94	86.72	86.92
MA_{10}	87.15	87.17	87.11	86.84	**87.06**

Table 4: Average attribute accuracy on YouTube Faces labeled test (T_0, T_1, T_2, T_3) using MA_i.

Model	T_0	T_1	T_2	T_3	Average
TC_1	86.52	86.63	86.60	86.40	86.53
TC_2	86.74	86.81	86.80	86.52	86.71
TC_3	86.77	86.85	86.88	86.54	**86.76**
TC_{10}	86.54	86.60	86.54	86.39	86.51

Table 5: Average attribute accuracy on YouTube Faces labeled test (T_0, T_1, T_2, T_3) using TC_i.

4.4. Motion-Attention

For our motion-attention experiments, we similarly fine-tune MACNN on anchor and non-anchor frames, using the labels from anchor frames for both. The motion-attention is applied after the first pooling layer in MACNN. MA_1 is the motion-regularized model trained on f_0, and f_1 with corresponding motion-attention maps m_{00} and m_{01} respectfully. That is, m_{00} is an image of all 1s because there is no motion before the anchor frame, and m_{01} is a binary image capturing the motion between f_0 and f_1. MA_1 uses a sigmoid cross-entropy loss for both anchor and non-anchor frames using attribute labels from anchor frames (i.e. both f_0 and f_1 are labeled with the attributes from f_0). MA_2 is the motion-attention model trained on f_0, m_{00}, f_1, m_{01}, and f_2 and m_{02} (f_0, f_1, and f_2 are labeled with the attributes from f_0), and so on. MA_0 is equivalent to $MACNN_0$, since m_{00} – defined to be all 1s – provides no attention. The average attribute accuracies on anchor frames using MA_i for $i = 1, 2, 3, 4$ are reported in table 4. There is a consistent improvement when fine-tuning with motion-attention, and as non-anchor frames are added to the training set, the improvements continue, unlike regular fine-tuning. The motion-attention mechanism keeps the model from overfitting since it cannot see the entire non-anchor frame, and so the motion-attention works as a kind of regularizer. We see that even MA_{10} shows improvements where $MACNN_{10}$ showed a sharp decline in performance.

Model	T_0	T_1	T_2	T_3	Average
$MATC_1$	86.60	86.75	86.68	86.41	86.61
$MATC_2$	86.93	86.98	86.98	86.61	86.87
$MATC_3$	87.04	87.05	87.01	86.71	**86.96**
$MATC_{10}$	86.83	86.79	86.66	86.43	86.67

Table 6: Average attribute accuracy on YouTube Faces labeled test (T_0, T_1, T_2, T_3) using $MATC_i$.

4.5. Temporal Coherence

For the TC loss experiments, we define $\lambda(t) = e^{\frac{1}{t}-1}$ for $t \geq 1$, so the effect of each non-anchor frame on learning decreases with time. When fine-tuning MACNN using the TC loss, the loss is employed between the final convolution layers (conv4) of the two streams. A sigmoid cross-entropy loss is applied to the anchor frames – to learn the attributes in the anchor frames – in addition to the TC loss. Figure 2 visualizes the attribute loss on the anchor stream, and the TC loss between the two conv4 layers. The figure only visualizes two streams, but there can be many streams depending on the number of non-anchor frames used in training. We call our models trained with the TC loss TC_1 if it fine-tunes on f_0 and f_1, TC_2 if it fine-tunes on f_0, f_1, and f_2, and so on. TC_0 is equivalent to $MACNN_0$, because without a non-anchor frame, there can be no TC loss, and so we end up with a single-stream network employing a sigmoid cross-entropy loss on the attribute labels of anchor frames. Table 5 shows the average attribute accuracy on the anchor frames over the 10 splits of YouTube Faces using the TC loss while fine-tuning. As seen with motion attention, there is a consistent improvement using the TC loss when fine-tuning, and as the number of non-anchor frames used for training increases, so does the average attribute accuracy. However, unlike MA_{10}, we do not see an improvement with TC_{10}. We believe this is due to the fact that the network is over-fitting a little, even with the TC loss. Though we do see a drop in performance with TC_{10}, it is not nearly as severe as the one we saw with $MACNN_{10}$, so the TC loss is helping the network to not overfit, and to account for the attributes which can change throughout a video. The TC loss on non-anchor frames improves over $MACNN_i$ because it is less strict, and therefore allows pairs of anchor and non-anchor frames to have different attribute labels.

4.6. Motion-Attention with Temporal Coherence

We combine the proposed motion-attention mechanism and temporal coherence loss into one network by applying both methods on non-anchor frames. Combining both methods results in a multi-stream network where the anchor frames use a sigmoid cross-entropy loss, and the non-anchor frames each employ a TC loss and a motion-attention mechanism. We apply the motion-attention after the first pooling layer, and the TC loss at the conv4 layer. The models trained in this way are denoted $MATC_1$ (one non-anchor frame), $MATC_2$ (two non-anchor frames) and so on. Table 6 reports the average attribute accuracy for each of the MATC models on the labeled frames. Combining the two methods results in an improvement over using them individually, with $MATC_1$, $MATC_2$, and $MATC_3$ producing better results than both TC and MA individually. We do see that the dip in performance for TC_{10} carries over to $MATC_{10}$, reducing performance slightly. We again believe this to be due to the network slightly overfitting even with the TC loss.

5. Conclusion

We introduced two methods for explicitly incorporating video information into the training of attribute networks: a temporal coherence constraint and a motion-attention mechanism. Though some work has been done on adapting attribute models trained on CelebA to better handle data from different distributions [26], we argue and show that time and motion must specifically be accounted for when training attribute models on video data. We demonstrated the effectiveness of our methods on the challenging YouTube Faces dataset, improving over the baseline of fine-tuning directly on the video data using only weakly labeled data. Our results show that when we do not account for time and motion in learning attribute models, as in $MACNN_i$, the model behavior is erratic. The proposed methods are able to use information provided by many non-anchor frames far away from the original anchor frames, which we cannot do when fine-tuning without these methods. We also note that there are only 3425 anchor frames, and so the fine-tuning is performed using roughly 3000 frames for each split, which is a very small amount of data for a CNN. As more labeled video data becomes available, the positive effects of the proposed temporal coherence and motion-attention methods will be even more obvious. Our results demonstrate the need to explicitly account for temporal and motion constraints when training attribute models on video data. The next step in learning robust attribute models for video data is to label a new video dataset with facial attributes, so that the effects of time and motion on attributes can be more thoroughly studied.

Acknowledgement

This research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA R&D Contract No. 2014-14071600012. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

References

[1] J. R. Barr, K. W. Bowyer, P. J. Flynn, and S. Biswas. Face recognition from video: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, 2012.

[2] H. T. Cheng, F. T. Sun, M. Griss, P. Davis, J. Li, and D. You. Nuactive: Recognizing unseen new activities using semantic attribute-based learning. *ICMSAS*, 2013.

[3] K. Duan, D. Parikh, D. Crandall, and K. Grauman. Discovering localized attributes for fine-trained recognition. *CVPR*, 2012.

[4] M. Ehrlich, T. J. Shields, T. Almaev, and M. R. Amer. Facial attributes classification using multi-task representation learning. *CVPR*, 2016.

[5] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. *CVPR*, 2009.

[6] Y. Fu, G. Guo, and T. S. Huang. Age synthesis and estimation via faces: A survey. *PAMI*, 2010.

[7] R. Goroshin, J. Bruna, J. Tompson, D. Eigen, and Y. LeCun. Unsupervised learning of spatiotemporally coherence metrics. *ICCV*, 2015.

[8] E. M. Hand, C. Castillo, and R. Chellappa. Doing the best we can with what we have: Multi-label balancing with selective learning for attribute prediction. *AAAI*, 2018.

[9] E. M. Hand and R. Chellappa. Attributes for improved attributes: A multi-task network utilizing implicit and explicit relationships for facial attribute classification. *AAAI*, 2017.

[10] S. J. Hwang, F. Sha, and K. Grauman. Sharing features between objects and their attributes. *CVPR*, 2011.

[11] D. Jayaraman and K. Grauman. Slow and steady feature analysis: Higher order temporal coherence in video. *CVPR*, 2016.

[12] D. Jayaraman, F. Sha, and K. Grauman. Decorrelating semantic visual attributes by resisting the urge to share. *CVPR*, 2014.

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint*, 2014.

[14] N. Kumar, P. Belhumeur, and S. Nayar. Facetracer: A search engine for large collections of images with faces. *ECCV*, 2008.

[15] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. *ICCV*, 2009.

[16] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Describable visual attributes for face verification and image search. *PAMI*, 2011.

[17] H. Liu, J. Lu, J. Feng, and J. Zhou. Two-stream transformer networks for video-based face alignment. *PAMI*, 2017.

[18] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. *ICCV*, 2015.

[19] H. Mobahi, R. Collobert, and J. Weston. Deep learning from temporal coherence in video. *ICML*, 2009.

[20] C. B. Ng, Y. H. Tay, and B. M. Goi. Vision-based human gender recognition: A survey. *arXiv preprint*, 2012.

[21] J. C. Niebles, C. W. Chen, and L. Fei-fei. Modeling temporal structure of decomposable motion segments for activity recognition. *ECCV*, 2010.

[22] H. S. Parekh, D. G. Thakore, and U. K. Jaliya. A survey on object detection and tracking methods. *International Journal of Innovative Research in Computer and Communication Engineering*, 2014.

[23] E. Park, X. Han, T. L. Berg, and A. C. Berg. Combining multiple sources of knowledge in deep cnns for action recognition. *WACV*, 2016.

[24] R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 2010.

[25] R. Ranjan, V. M. Patel, and R. Chellappa. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *arXiv preprint*, 2015.

[26] E. Rudd, M. Gunther, and T. Boult. Moon: A mixed objective optimization network for the recognition of facial attributes. *ECCV*, 2016.

[27] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human action classes from videos in the wild. *CRCV-TR-12-01*, 2012.

[28] M. Toews and T. Arbel. Detection, localization, and sex classification of faces from arbitrary viewpoints. *PAMI*, 2009.

[29] J. Wang, Y. Cheng, and R. S. Feris. Walk and learn: Facial attribute representation learning from egocentric video and contextual data. *CVPR*, 2016.

[30] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. *ICCV*, 2015.

[31] L. Wolf, T. Hassner, and I. Maoz. Face recognition in unconstrained videos with matched background similarity. *CVPR*, 2011.

[32] N. Zhang, M. Paluri, M. A. Ranzato, T. Darrell, and L. Bourdev. Panda: Pose aligned networks for deep attribute modeling. *CVPR*, 2014.

[33] Z. Zhang, P. Luo, C. Loy, and X. Tang. Facial landmark detection by deep multi-task learning. *ECCV*, 2014.

[34] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *CSUR*, 2003.

[35] J. Zheng, Z. Jiang, R. Chellappa, and J. P. Phillips. Submodular attribute selection for action recognition in video. *NIPS*, 2014.