Comparative Graph Model for Facial Recognition Systems

Timothy Zee, Mehdi Ghayoumi

Kent State University

Overview

Comparative Graph Model is a facial recognition algorithm that takes images of a human face, and identifies common features within the training data for that face. The algorithm looks at the closeness of features within images, along with how many times each exists, to decide which features are the best representation of that face. This algorithm is a different approach than many facial recognition algorithms because it does not use a template to seek out facial features. CGM uses only data that is in the training set to form features. This makes the algorithm completely unsupervised.

In this experiment, CGM is used in conjunction with a perceptron neural network to compare positive and negative matches. Ten training images were condensed to one image via CGM, and compared against one positive image (the same person), and one negative image (a different person). In the experiment, each trial was run for 100 iterations. The algorithm recognized the trained person 71% correctly, did not recognize a new person (someone not from the training set) 79% correctly, with an overall correctness of 61%. Since CGM is unsupervised, the algorithm is beneficial across many disciplines, especially where a pattern can be too hard to find or represent. The algorithm does not need to know what it is looking for to make associations, which makes it a good candidate for finding complex patterns and associations in data.

Background

Feature

A feature is a distinct point found within an image. Features are used to make comparisons between images. To find features, algorithms such as SURF (Speeded-Up Robust Features) or SIFT (Scale-Invariant Feature Transform) can be used. Features are often found at corners in lines, or at large slope changes within lines.

Feature Map

To compare entire faces to one another, individual features are not very helpful. It is how the features are relative in position to other features that is valuable. Facial position will vary due to variables such as pose, position, size, etc. However, distance ratios between features are more resilient to these variables. The ratio between various features will remain the same, even at different positions. Feature maps are compared to find similarities in a training set.

Perceptron Neural Network

Artificial Neural Networks (ANNs) are popularly used for machine vision and recognition. ANNs are biologically-inspired algorithms based on the human brain's ability to learn. ANNs use computational representations of neurons to receive signals through synapses located on the dendrites, or the membrane of the neuron. This is represented in single or multi-layered networks that send data through several neurons that changes the weights of input to compute a result. Once trained, new test data can be fed into the network producing positive or negative results. A perceptron neural network is a specific type of ANN that uses a one-level deep network.

Template-Based Algorithm

Template-based algorithms are supervised algorithms that require a template for data to be fit into. Features are associated with specific parts of the template. This is a popular technique for facial recognition, as well as facial tracking.

Comparison

Comparative Graph Model:

Positives	Negatives
▪ Completely unsupervised ▪ Gives all found features equal opportunity ▪ Finds all distinct features within the training set	▪ Much slower due to comparative nature of the algorithm ▪ Currently cannot be used for real-time systems

Figure 1. Listing benefits and drawbacks to Comparative Graph Model.

Template-Based Approach:

Positives	Negatives
▪ Very fast algorithm ▪ Works relatively well for most applications	▪ Requires a template of the object being sought ▪ Supervised ▪ Unique facial features can be missed

Figure 2. Listing benefits and drawbacks to a template-based approach.

Algorithm

Comparative Graph Model decides on important features by examining how close features are to one another throughout the training set, and how many times features exist. By reviewing all features, CGM makes decisions on whether or not a feature accurately represents the person. CGM outputs a map of points that are the best representation of a person's face. All features that meet the criteria will end up in the final map of features.

Preprocessing

The first step of CGM is to preprocess the data to find features in the image set. First, all training data is turned to gray scale, and an edge detection algorithm is applied. In this study, a Sobel edge detector is used. By creating an edge map rather than the whole image, it is easier to identify features in a facial image. The last step of preprocessing is to take the images with only edges and condense them into a number of features (points). This makes feature comparison possible. A SURF detector is used in this experiment.

Setting the Allowable Distance Threshold

Features in one image should align with features in other images if those features are a good representation of the data. Therefore, those features should be found across images in the data set. Determining an allowable amount of distance from one feature in one image, to another feature in another image is vital. If the allowable distance is too small, any variation in images will not include these features. However, if the allowable distance is too large, features that are separate features could be considered the same feature and misrepresent data. We will represent this value as the Allowable Distance Threshold. In order to measure distance between features we use Euclidian distance. The percentage of allowable distance will be referred to as ϕ and the actual distance is referred to as δ. This can be found by the following:

$$\delta := \frac{\phi}{100n} \sum_{i=1}^{n} \sqrt{(Image_i x_2 - Image_i x_1)^2 + (Image_i y_2 - Image_i y_1)^2}$$

Figure 3. Equation to compute the Tolerable Distance Threshold.

Setting the Z Parameter

Now that we can find if two features in two images are the same feature, there is the need to set a minimum of how many times a feature must be found before it is considered a feature in the final node map. This value is referred to as the Z parameter. The tolerable distance threshold and Z parameter have a large impact in the algorithm's accuracy. If the Z parameter is too small, then false matches will exist, resulting in false features. However, if the Z parameter is too large, not enough accurate facial nodes will meet the requirement, and too small of a node map will be created, causing poor performance.

Finding Features that Appear Multiple Times

Now that there is criteria to determine if two features are the same point, each image must compare every feature to every other feature in all other images. The first feature is referred to as the original point, and the second is the test point. From the SURF detector, there is a list of points from each image. If the original feature and test feature are within the tolerable distance threshold, they are the same node. To store this information, a list that contains a list of points is used. If two nodes are found within the tolerable distance threshold, then the original point is added to the list of points that is the same index as the input index for that image. The test feature is added to the same index as it is in the input index for the test image. This keeps the same features in the same image feature indices as the input. The result of this step gives all features that have at least one match to some other feature in a different image. All points that do not have a match are eliminated.

Finding Occurrences of Features

Now all features have at least one match in the image set, but it is not known how many times a feature exists. A new data type is created that contains a point, and an amount of occurrences. This data type is called a master node. A list of master nodes is created to contain all of the information needed. Every feature in all images is now compared to every other feature in all other images. If an original and test feature are within the tolerable distance threshold, then it is checked if the original or test feature is already in the master node list. If it is, increment the occurrences of that feature. If it is not, add the original feature to the master node list. Now only distinct features will exist and will have a count of how many times that feature exists within the data.

Allowing Only Features that Are Z% or Greater

Now the occurrences of each feature must be at least Z% of the training images to be in the master node list. Nodes that are not are removed. All features now have met both parameters, and are the feature map of the training set.

Trimming Features to Match the Test Image

In order to test positive and negative matches, every test image goes through step A. Now take the minimum amount of features of the feature map and the test image. This will be the amount of nodes each image will have after trimming. To trim features, find the closest match between the training and test image, and use that. Then save this feature and continue the process for the next closest image. Continue this until the minimum number of features of the training image and test image has been found.

Using a Perceptron Neural Network

Finally, a perceptron neural network can be used to determine if the test image is the same person as the training images, or if it is a different person.

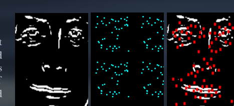

Figure 4. After a Sobel Line Detector and SURF Detector have been utilized, CGM displays the filtered result.

Experimental Results

Comparative Graph Model needs a tolerable distance threshold and an amount of images to match across a (known as Z). These two parameters allow the algorithm to have an optimum configuration for accuracy based on the domain that it is used in. For facial recognition, we have tested the tolerable distance threshold from 1% – 4% with quarter percent intervals. Z is tested at 40% – 70% in ten percent intervals. This gives us an overview of where the optimal values for each parameter could be. Each tolerable distance threshold and Z are tested at 100 iterations. This allows us to find the most accurate configuration of this algorithm for facial recognition using a perceptron neural network.

In this dataset, we took 10 front-facing images of the same person for our training set, and then one additional image to use to verify the same person, and an image of a different person, that was not trained, to test false matches. The resulting overall accuracy for Z = {40, 50, 60, and 70} with each allowable distance threshold of TDT = {1.00, 1.25, 1.50, 1.75, 2.00, 2.25, 2.75, 3.00, 3.25, 3.50, 3.75, and 4} are shown in the table as follows:

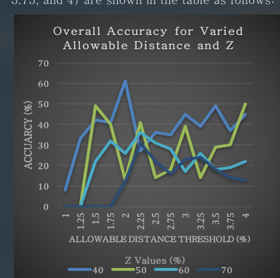

Figure 5. A graph showing the resulting accuracies at various TDT and Z choices.

As seen in the diagram, the Z parameter directly affects where the optimal value is found. As the Z parameter becomes more restrictive, the optimal point is shifted more right (where TDT allows for a larger gap between nodes). Thus, TDT and Z have an inverse relationship to one another. As one parameter becomes more restrictive, the other parameter needs to become less restrictive to find an optimal accuracy.

Therefore, there exists a balancing act between these two parameters that will create the optimal value for the algorithm. This study shows that the optimal value may be found around TDT = 2.00% and the Z parameter = 40%. At this point, the overall accuracy achieved was around 61% with a perceptron neural network. As either parameter becomes too restrictive, or not restrictive enough, overall accuracy begins to diminish.

Future Work

Future work for Comparative Graph Model includes improvements in a few areas:

Efficiency

Comparative Graph Model can benefit from improved efficiency by reducing the amount of comparisons computed. If features all belonged to specific regions, regions can be compared to the same regions across other images. This would reduce the amount of computation significantly. To ensure features near the border of regions are not missed, a region can be compared to the same region plus the first neighboring feature (in all directions) across all other images in the training set.

Figure 6. By comparing regions against the same regions (plus neighbors) in other images, the amount of comparisons needed is significantly reduced.

Optimized Parameters

The tolerable distance threshold and Z parameter in this study were tested at different values to manually find optimal values. The next iteration includes finding these optimal tolerable distance threshold and Z parameters automatically before the algorithm runs.

Convolutional Neural Network

The results achieved from this study were done with a one-layer perceptron neural network. Perceptron neural networks cannot achieve more than 80% – 85% accuracy. In order to truly test how accurate CGM can be, using a more accurate machine learning technique needs to be used. The next step is to use a convolutional neural network that can achieve up to 98% – 99% accuracy to see how accurate CGM can be for the use of facial recognition and authentication purposes.

Conclusion

Comparative Graph Model is able to use all distinct features found within the training set. This method allows for a fuller node map than template-based approaches. This study introduces the CGM concept and some key areas where it may be beneficial compared to other approaches. Overall, in this experiment, the algorithm was able to achieve an overall accuracy of 61%, with 71% correctness of matching a person to themselves, and 79% of not matching a different person to the trained person. This was found at a tolerable distance threshold of 2.00% and Z at 40%.

By using a deep learning algorithm, such as a convolutional neural network, CGM will not be bounded by the limited correctness possible within a perceptron neural network (around 80% – 85% correctness). As shown by the experimental results, the resulting accuracy is a balancing act of the tolerable distance and Z parameter. With a careful choice of these parameters, this algorithm shows viable results, and with continuing efforts, may be a useful algorithm for discovering patterns in training data due to the unsupervised nature of the algorithm.

Timothy Zee, Author

E-mail: tzee@kent.edu

GitHub: https://github.com/tzee

Mehdi Ghayoumi, Advisor

E-mail: mghayoum@kent.edu