Gaussian Mixture Models and Image Upscaling

Spencer A. Hammersten
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA 56267
hamme503@morris.umn.edu

ABSTRACT

Image super-resolution is a concept with important applications in law enforcement, digital entertainment, medicine, and many other fields. The primary focus of this paper is to examine and explain the use of Gaussian Mixture Models (GMMs) to aid in increasing the resolution of images. The GMMs are trained via dictionaries of image pairs with a low- and high- resolution version of the same image patch. This paper is a deep dive on a research paper by Dongfeng Meia et al [10], and it describes in detail the method applied by those researchers. To do this, several statistical concepts are covered, such as Gaussian distributions, regression, and mixed models.

1. INTRODUCTION

There is some disagreement among authorities regarding the definition of the term "super-resolution". To some, it functions simply as a synonym for "upscaling", which is the process of increasing the size (resolution) of an image. To others, it is only applicable as a term if the smaller (low-resolution, or LR) version of the image has never had a larger (high-resolution, or HR) counterpart. For the purposes of this paper, I use the terms interchangeably.

Image super-resolution serves many purposes and an efficient, automated method of detail restoration and enhancement would have wide-reaching implications throughout numerous fields. For example, details could be recovered from low-resolution crime scene photos that help identify suspects, or such a technology could be used to increase the visual fidelity of films, video games, and other forms of visual entertainment. Due to the broad incentives, this technology has received substantial attention.

Convolutional neural networks have been shown to accomplish image-related tasks very effectively [13], super-resolution being no exception [5]. Another type of machine learning applied commonly to image super-resolution is called sparse dictionary learning; this is an effective way of performing super-resolution on an image when that image can be easily divided into patches with specific details, such as images taken from nature. The paper [10] documents several successional attempts to use this type of machine learning for super-resolution, each solving the issues of a previous attempt.

UMM CSci Senior Seminar Conference, April 2019 Morris, MN.

The paper [10] then presents a new method: using Gaussian Mixture Models (GMMs), a statistical model for describing multiple trends within a set of data, to classify image patches into categories. These categories can then be used to create multiple pairs of something called dictionaries representing what is called a sparse representation of an image patch category (these terms are explained later). Using the GMM-classified categories, the dictionaries can represent more minute details, and the result is an upscaled image with not only better defined large features, as is easy to do with sparse dictionary learning, but also better defined details.

The research in question was performed by Dongfeng Meia et al[10]. The approach uses many of the same techniques as the machine learning-based methods, but using image patches, allows different portions of an image to be processed in different ways by an algorithm or model, allowing for more efficient training and more specific details to be recovered.

2. BACKGROUND

2.1 Composition of Images

A pixel is an atomic component of an image which represents a single point of color data. The majority of images are formatted as a collection of pixels. The resolution of an image is the number of pixels that make up a rectangular image, represented as a product of its dimensions. For example, a square image with 400 pixels total would have a resolution of 20x20. The higher an image's resolution, the more detail is visible within the image. The goal of most research into super-resolution is to increase the resolution while adding an appropriate level of detail for the target resolution. It is a difficult task, as the information required to enhance detail is usually not discernible from a low-resolution (LR) image.

The RGB system is the most common way of defining the color of a pixel, but for this method of super-resolution it is more useful to look at an image as a combination of luminance and chrominance data. These are sets of data that define the brightness of color throughout the image, and the hue, divided into two channels, throughout the image. This is the method in which video signals are broadcast over analog television [8]. The luminance channel is essentially a black-and-white version of the image, and the two chrominance channels are collections of hue data.

The luminance channel is denoted Y, and the chrominance channels are denoted CR and CB. The CR (chrominance-red) channel is comprised of the red value subtracted from

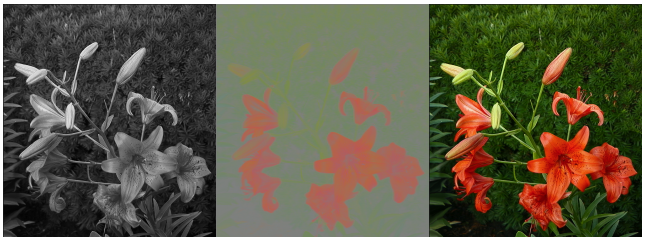

Figure 1: From left to right: The luminance channel, the chrominance-red channel, the full image [1]

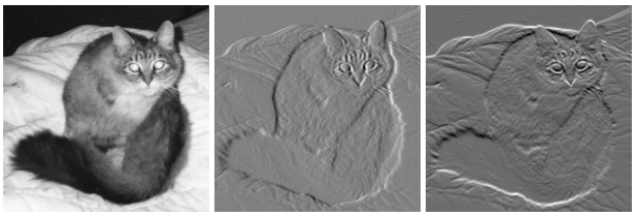

Figure 2: From left to right: the original greyscale image, the horizontal gradient, the vertical gradient [11]

the luminance data, and the CB (chrominance-blue) channel is comprised of the blue value subtracted from the luminance data. Humans are considerably more sensitive to luminance data than to chrominance data [10], so it is more efficient to perform computationally expensive super-resolution techniques on the luminance data alone, and use less intensive techniques to upscale the chrominance data. The method outlined in [10] makes use of this physiological characteristic.

An image patch is a section of an image that is usually small and rectangular. Typically, image patches are selected to be identical in shape and size. Dividing an image into patches is useful for the purpose of performing super-resolution; it can allow a program, algorithm or other computational entity to isolate the features of one specific area of the image and find an HR equivalent without needing to consider the rest of the image.

The gradient of an image's luminance is defined as the directional change in luminance value; it is the derivative of the values over a given direction in the image. Taking the gradient of an image is useful for isolating the overall features, or contours, of the image, as can be seen in Figure 2. This is taken in a single direction, isolating each line of pixels as though it were a linear equation, and determining the change in luminance between each consecutive pair of pixels. The second-order gradient is the gradient of the gradient. It is similar to the second derivative of a function.

2.2 Interpolation

Interpolation is the process of finding data points that lie in between a set of known data points. For the purposes of image upscaling, this means creating additional pixels with which to "fill in the gaps" between the original pixels. There are a substantial number of types of interpolation, but the most common are nearest neighbor, linear, and cubic.

Nearest-neighbor interpolation is the simplest; it assigns each pixel a color according to the color of the nearest existing pixel. This, in effect, causes the original appearance to be preserved exactly, only at a larger scale. Nearest-neighbor interpolation provides no smoothing effect and results in the presence of aliasing and jagged edges (a pixelated appearance, shown in Figure 4).

Linear interpolation defines a linear relationship between two data points, and fills in the points between the given data based on the corresponding value on the linear function. The linear function on a one-dimensional set of data is as follows, wherein, for our purposes, let us consider only two data points, x and y:

$$\frac{y - y_0}{x - x_0} = \frac{y_0 - y_1}{x_0 - x_1}$$

In this model, x represents the distance between the two data points and y represents the color as defined on an RGB scale.

Cubic interpolation utilizes a similar concept, but instead of defining a linear relationship based on two data points it instead defines a cubic relationship based on four data points. By utilizing more data points, it creates a smoother transition between values.

However, this process only determines data points on a

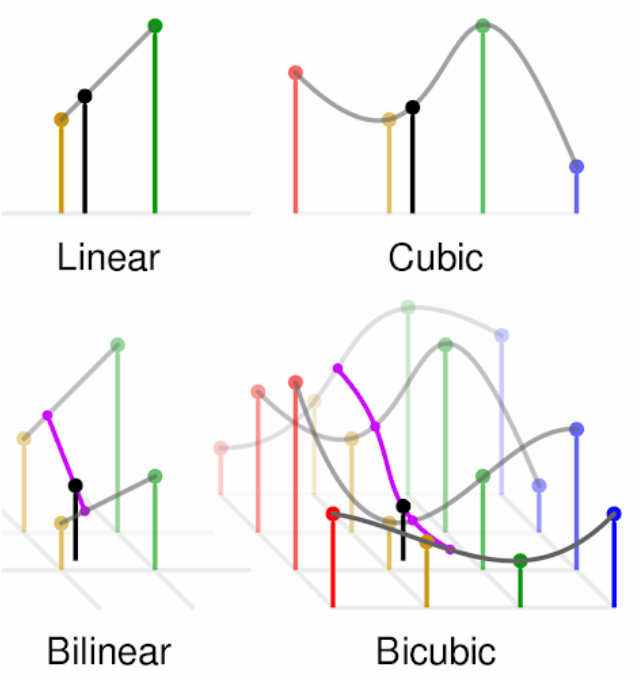

Figure 3: Graphs of the data points as calculated during interpolation [4]

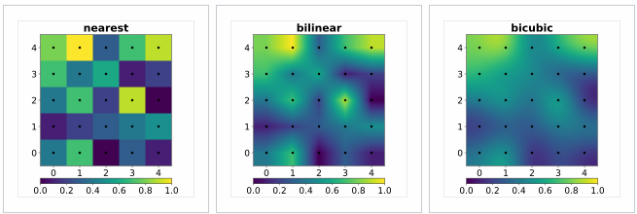

Figure 4: Comparison of the results of interpolation methods [17][18][16]

single line; in order to work with a fully two-dimensional image we need to use bilinear or bicubic interpolation. Bilinear interpolation, when used for upscaling, defines an interpolated relationship between two parallel sets of two points, then for each relevant point on each defined line, defines another linear relationship. A similar approach is used for bicubic interpolation: using four sets of four parallel points, cubic interpolations are drawn between each, and then drawn for each relevant point in the perpendicular direction. The result is an image with smoother value transitions between established data points, making the image more visually appealing but also making it more difficult to distinguish the original values of the given data points. A visual representation of the lines can be seen in Figure 3.

2.3 Linear combination

A representation of a data set is simply a set of vectors that can be linearly combined to recreate any member of that data set. To understand this it is useful to take a step back and think of vectors in two-dimensional space. It is common to think of vectors as arrows on a 2D plane; this is useful for understanding the concept of a linear combination. A vector on a 2D plane can be split into two components: an x component, and a y component. The x component describes the horizontal distance covered by the vector, and the y component describes the vertical. If one were to take two vectors, for the sake of argument take $(2x + 3y)$ and $(6x + 0y)$, one could combine them by adding together their x components, and then adding together their y components. The vector formed by these sums can be called a linear combination of them: $(8x + 3y)$. Let us label the first vector v, and the second vector w. A visual representation of this can be seen in the top half of Figure 5.

However, this is not the only linear combination of the two vectors that is possible. In linear algebra, a linear combination is not only the combination of the two vectors, but also any possible constant multiple of the two vectors. For example, we can take our second vector $(2x + 3y)$ and multiply it by a scalar (a constant): $2(6x + 0y)$. Then we add the two vectors once more: $(2x + 3y) + 2(6x + 0y) = (14x + 3y)$. This is also considered a linear combination of the two original vectors, and it can be notated as $v + 2w$. A visual representation of this is shown in the bottom half of Figure 5. We can do this with any constant, including negative constants and including zero constants. For a given set of vectors, the set of all possible vectors that can be created via linear combination of those is called the set's 'span'. This set of vectors, which is usually finite, forms what is called a 'basis', with which we can define a 'space'. Any element of a space can be represented as a unique linear combination of all of the vectors that form the basis for that space.

In two-dimensional space, any two vectors wherein one is not a constant multiple of the other span the entirety of the two-dimensional plane on which they exist. However, as we consider higher-dimensional vectors, spanning the entirety of a space becomes considerably more difficult. In fact, we need at least n vectors, none of which are constant multiples of one another, in order to span a space of n dimensions. Sometimes, though, our goal is not to span the entirety of an n-dimensional space, but instead to span a particular subsection of that space.

The most common way to represent a two dimensional plane is with two vectors $(x + 0y)$ and $(0x + y)$. These are

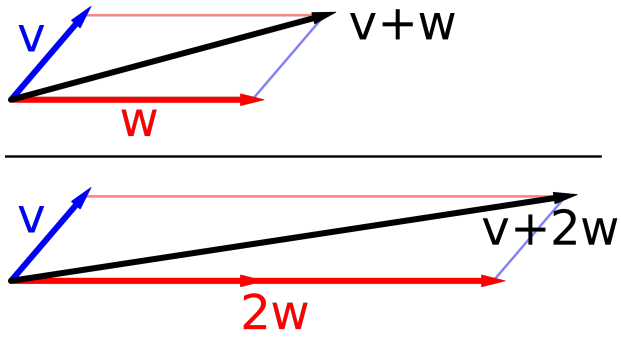

Figure 5: Visual representation of linear combinations [9]

our standard coordinates for naming a point in 2D space. Notice that in both vectors, one element is zero; this makes computations involving the vectors very easy, and it is why it is easy to think about two-dimensional space in terms of them. A representation of a data set where many elements of the representing vectors are zero is called a sparse representation of that data.

2.4 Sparse Dictionary Learning

For the purposes of this method, we can think of the luminance data of an image as a vector. Take each pixel as a component (such as x or y), and consider the image of n pixels as a vector in n-dimensional space. This is impossible to visualize as an arrow, but the mathematics are the same as they are in two dimensions. Much like in two dimensions, we can create a linear combination of these image vectors by adding together the luminance values of each pixel, or multiplying each by a constant before performing this sum.

A dictionary is a set of vector representations of a particular data format that can represent the entirety of a given set of input data. Let us consider the data format in question to be black-and-white images or image patches. In this case, a dictionary would be composed of a set of image vectors, structured as described in the previous paragraph, which span (can be linearly combined to represent any member of) a given set of input images. The goal of sparse dictionary learning is to find a dictionary that provides a sparse representation of the given input space. The members of the dictionary are referred to as 'atoms'.

A sparse representation of a set of images would be one in which the majority of the pixels of our vector images are completely black. Most of the time this isn't possible, so we frequently settle for very dark pixels instead of completely black ones. A sparse representation is useful for representing the contours of a type of image patch with as few vectors as possible. An effective method of doing this is to take the gradient of the image patch, which results in what's called a 'feature vector'. These feature vectors are much sparser than most image patch vectors because areas in which the luminance data of the image is relatively consistent become mostly zero.

In machine learning, training is the process of providing a learning algorithm with input data and penalizing results that do not match the desired outcome. A common method of training a sparse dictionary is an algorithm called K-SVD. This algorithm uses what is called singular-value decompo-

sition upon the dictionary atoms. Singular-value decomposition is a method of factorization for a matrix (a finite set of vectors). The algorithm finds a sparse representation by alternating between two steps. The first is to arbitrarily select a subset of atoms and form an over-representational dictionary for the input space. The second is to individually update the atoms in the dictionary so as to better represent the input space. A more detailed outline of this algorithm can be found in [12].

For the purposes of image super-resolution, a common application of this is to train a pair of dictionaries: a set of HR images or image patches, and downsampled or degenerated LR equivalents of the same image patches. By training a dictionary that represents a particular image patch, a linear combination of LR training data can be found that represents the input image patch. Using that combination, we can find an identical linear combination of the HR patches that are analogous to the LR patches that were combined. This approximates a HR equivalent to the input image patch. A more in-depth explanation of this process can be found in [15].

2.5 Application of GMMs

The method described in [10] makes use of these techniques, but in addition, it makes use of the training multiple pairs of dictionaries for several distinct categories of image patch. These categories are obtained by use of a Gaussian Mixture Model (GMM) and the expectation-maximization (EM) algorithm. The classification is performed using probabilistic analysis; using the aforementioned visual properties, the EM algorithm can predict which category a particular image patch falls into.

2.6 Gaussian distribution

A Gaussian, or normal, distribution is a model that describes a set of data points that is shaped like a bell curve; the majority of data points fall towards the center of the output range, while a smaller number are outliers. The mean of a population density curve is represented by μ, and the standard deviation by σ. The variance is the standard deviation squared. The standard normal distribution graph has a variance of 1, as shown in the red curve in Figure 6. The other curves shown in Figure 6 have differing mean and variance. The curves are called probability density functions and they are used to predict with some degree of certainty the likelihood of a data point being in a particular range.

Gaussian distributions can be multivariate, meaning that they are described by a set of single-variable Gaussian distributions relating to two or more variables. An example of this can be seen in Figure 5. In this case, the probability density curve is essentially three-dimensional, and the data is often most easily represented in image form by an ellipse with a centered gradient color denoting the variance of the distribution. The green ellipse filled with the black data points in Figure 7 creates a similar effect without the representational use of a third dimension.

2.7 Gaussian Mixture Models

A Gaussian Mixture Model is a probability density function that is described by more than one Gaussian distribution. The distributions are referred to as components of the model, and each has a mean and a variance. A single-variable example of a GMM may in fact look very similar

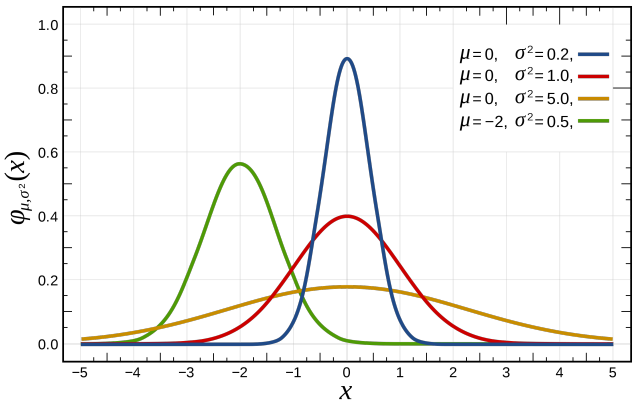

Figure 6: Several data curves with gaussian distributions. The red curve is the standard normal distribution with mean 0 and variance 1. [7]

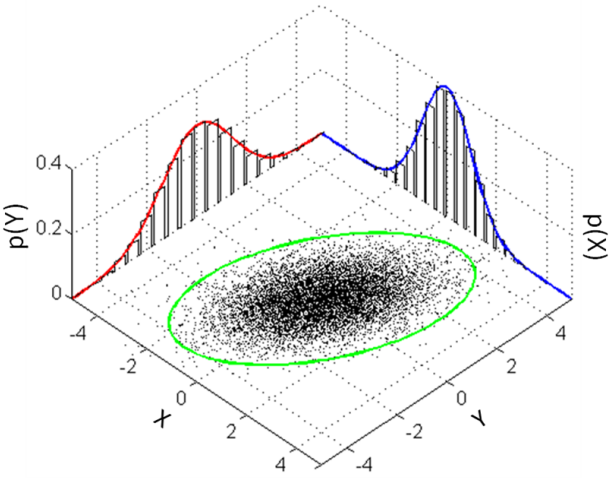

Figure 7: An example of a multivariate Gaussian distribution. [3]. p(x) and p(y) refer to probability density functions of variables x and y respectively.

to the curves shown in Figure 6. Using this type of model, it is possible to make statistically-based categorizations of data points based upon the likelihood that they fall within a particular component of the model.

GMMs can also be single-variable or multivariate. Multivariate GMMs are what the research utilizes for categorization of image patches. The process of categorizing the data points of the GMM and forming a probabilistic basis for categorization of future extracted image patches is referred to as training the GMM, and it is done via the EM algorithm. After training, the multivariate GMM can be referenced to predict the likelihood that an image patch falls within a particular component, after which the rest of the procedure can be performed on the individual image patches.

2.8 EM Algorithm

The EM algorithm is an algorithm that, via the use of two distinct steps, produces Gaussian components from a data set. It is useful for situations in which there are currently unknown data points that are expected to exist. The algorithm alternates between an expectation step and a maxi-

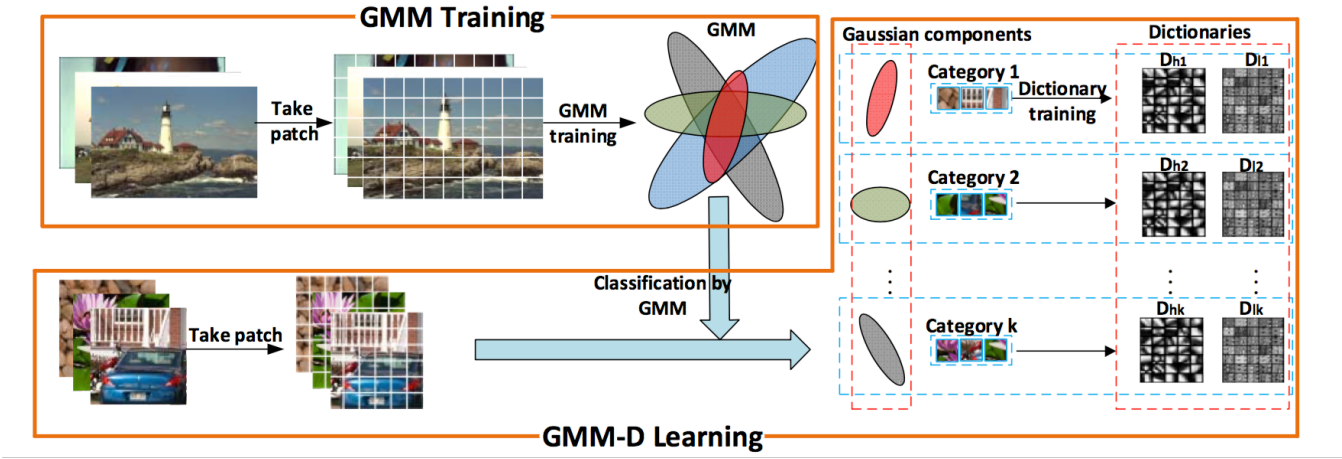

Figure 8: Visualization of the GMM training process [10]

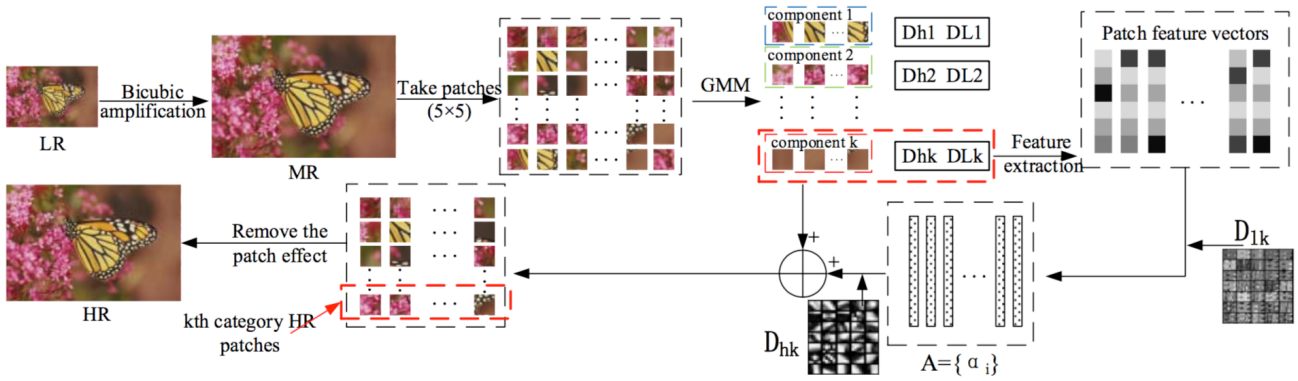

Figure 9: Visualization of the super-resolution process on an input image [10]

mization step. The expectation step estimates a set of components to attempt to fit an arbitrarily selected portion of the data. The maximization step introduces more of the data and maximizes the likelihood of the parameters being accurate based on the new data. The process repeats until the maximum likelihood is the same as the estimated parameters. A more detailed explanation of this algorithm can be found in [2].

3. THE GMM METHOD

The process outlined in [10] begins with the training of a Gaussian mixture model with some number of components, K. [10] does not specify how K is found, but there are algorithms that exist for determining the number of Gaussian components to define for a set of data. For training this GMM, patches are extracted from a sizable set of "natural images" [10], and using the data contained within those patches, GMM components representing the data can be extracted. The data points in question are luminance and chrominance values of pixels obtained from the original images. The paper does not go into meaningful detail regarding exactly how the data points are represented for the purposes of data analysis. The visuals provided by the paper suggest the categories are likely to depend upon what exactly is being represented in the patch: sky, grass, building walls, fur, etc.

Following the creation of image patch categories by GMM, a new set of HR natural images is used to create multiple pairs of dictionaries. Patches from these training images are extracted and classified according to the GMM components discovered in the first step. The following steps are performed upon each category of image patches. The second-order gradient of the patches' luminance data is extracted and used, as a feature vector, to train a HR dictionary with the purpose of sparsely representing the given image patch category. In order to attain a pair of dictionaries, the HR patches are downsampled (lowered in resolution by sampling evenly spaced pixels throughout the image) and degenerated using a fuzzy effect to obtain LR versions of the same patches. A visual representation of the process up to this point can be seen in Figure 8.

In order to perform super-resolution on a given input image, the input image is first upscaled to a mid-resolution (MR) via bicubic interpolation. This MR image is then broken up into patches, which are classified according to the

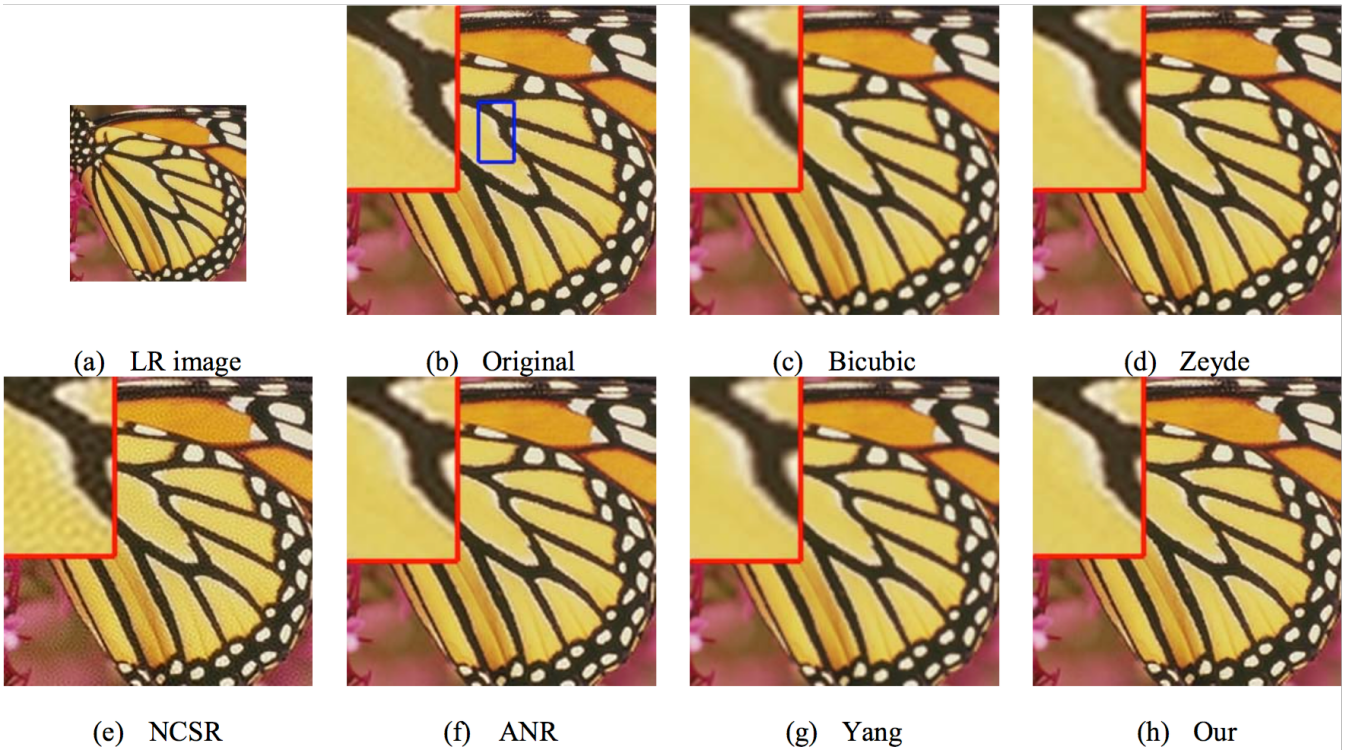

Figure 10: Comparison of results of various methods of super-resolution [10]. "Our" refers to the result of the method outlined in the [10].

GMM trained in the first step of the training process. At this point the chrominance data is upscaled to the target resolution via bicubic interpolation, to be added back in at the very end. The following steps are executed only on the luminance data. The LR dictionary finds a linear combination to approximately represent the input patch, which is then combined with the MR luminance data of the original patch to get the final HR output patch. Finally, the patches are combined together, the patch effect is removed (the paper does not specify the methods used for this) and the chrominance data is added back in after bicubic interpolation. A visual representation of this process can be seen in Figure 9.

4. RESULTS

The paper [10] compares the results of a variety of methods of super-resolution, as shown in Figure 10. It shows the original, interpolated upscaling versions, a few of the other sparse dictionary learning methods including [5], [14], and [6], and the GMM method.The GMM method is meaningfully more detailed than the other methods, and re-creates the contours of the original HR image a bit more accurately without introducing extraneous detail like, for example, the NCSR [6] result does. The paper [10] describes it as more visually appealing, which, while not the exclusive goal of image super-resolution, is an important component of the process.

5. CONCLUSIONS

The paper in question demonstrates the usefulness of GMMs for the purpose of image super-resolution. Through the use of GMM Training, multi-pairs of dictionaries for sparse learning, and interpolation on a set of "natural images", an effective method of automated super-resolution can be achieved. The paper demonstrates the benefits of this method by comparing it to various other algorithms intended to achieve the same goal, and showing visually the advantages of the newly proposed method.

Acknowledgements

I'd like to extend my sincerest gratitudes to Professors Peter Dolan and Elena Machkasova, without whom this paper quite literally could not exist. They have assisted me through this process and displayed tremendous patience, understanding, and kindness in addition to the consistent feedback. I'd also like to thank Humza Haider, my external alumni reviewer who provided me with additional feedback on my paper.

6. REFERENCES

[1] Algr. *Orange flowers w/ green background, presented in three side by side images showing luminance only, chroma only, and full color.* Wikimedia Commons, Feb 2014.

[2] S. Borman. The expectation maximization algorithm-a short tutorial. 2004.

[3] Bscan. *Illustration of a multivariate gaussian distribution and its marginals.* Wikimedia Commons, Mar 2013.

[4] Cmglee. *Comparison of nearest-neighbour, linear, cubic, bilinear and bicubic interpolation methods by*

CMG Lee. The black dots correspond to the point being interpolated, and the red, yellow, green and blue dots correspond to the neighbouring samples. Their heights above the ground correspond to their values. Wikimedia Commons, Nov 2016.

[5] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 184–199, Cham, 2014. Springer International Publishing.

[6] W. Dong, L. Zhang, G. Shi, and X. Li. Nonlocally centralized sparse representation for image restoration. *IEEE Transactions on Image Processing*, 22(4):1620–1630, April 2013.

[7] Inductiveload. *A selection of Normal Distribution Probability Density Functions (PDFs).* Wikimedia Commons, April 2008.

[8] A. Kaiser. Chrominance-luminance separator, Feb. 7 1978. US Patent 4,072,984.

[9] kamusumeFan. *Vector addition and scalar multiplication illustration.* Wikimedia Commons, Jul 2015.

[10] D. Mei, X. Zhu, C. Yue, Q. Cao, L. Wang, L. Zhang, and Q. Song. Image super-resolution based on multi-pairs of dictionaries via patch prior guided clustering. In *2018 Eighth International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6, Nov 2018.

[11] Njw000. *On the left, an intensity image of a cat. In the center, a gradient image in the x direction measuring horizontal change in intensity. On the right, a gradient image in the y direction measuring vertical change in intensity. Gray pixels have a small gradient; black or white pixels have a large gradient.* Wikimedia Commons, Jun 2010.

[12] R. Rubinstein, A. M. Bruckstein, and M. Elad. Dictionaries for sparse representation modeling. *Proceedings of the IEEE*, 98(6):1045–1057, 2010.

[13] I. Sutskever, G. E. Hinton, and A. Krizhevsky. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, pages 1097–1105, 2012.

[14] R. Timofte, V. De, and L. V. Gool. Anchored neighborhood regression for fast example-based super-resolution. In *2013 IEEE International Conference on Computer Vision*, pages 1920–1927, Dec 2013.

[15] J. Yang, J. Wright, T. S. Huang, and Y. Ma. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing*, 19(11):2861–2873, Nov 2010.

[16] Zygure. *Illustration of Bicubic interpolation on a random dataset.* Wikimedia Commons, Sep 2016.

[17] Zygure. *Illustration of Bilinear interpolation on a random dataset.* Wikimedia Commons, Sep 2016.

[18] Zygure. *Illustration of en:Nearest neighbor interpolation on a random dataset.* Wikimedia Commons, Sep 2016.