2018 IEEE International Conference on Big Data and Smart Computing

Image Analogy with Gaussian Process

Chan-Yong Park, Han-Gyu Kim, Dong-Keon Lee, Zhun Li, Seung-Ho Han, Ho-Jin Choi
School of Computing, Korea Advanced Institute of Science and Technology (KAIST)
291 Daehak-ro, Yuseong-gu, Daejeon, Republic of Korea
{*ptparty, kimhangyu, hagg30, lizhun, seunghohan, hojinc*}*@kaist.ac.kr*

***Abstract*—Image analogy is the process of creating an image filter that precisely reflects the characteristics contained in the training data. Recently, the image analogy problem was generally handled by deep neural network (DNN) with the development of a deep learning technology. Generally, DNN suffers from a fatal problem in that it requires large amounts of data for training. However, as pairs of images with the same relationship are needed for an image analogy, it is hard to collect sufficient data for image analogy using DNN. In order to solve this problem, we propose an image analogy method using a Gaussian process. In this method, a Gaussian process regression is used instead of DNN regression to adjust the feature vectors which will be used in creating filtered image. Additionally, in order to accelerate the training speed of Gaussian process, we also propose novel sampling methods that select salient instances from a given dataset. Our experiment result demonstrates that the proposed image analogy method using a Gaussian process with salient instance sampling performs significantly better than DNN in environments with small dataset size.**

***Keywords*-image analogy, image process, Gaussian process, sampling.**

I. Introduction

Image analysis can be divided into two parts. The first part is analyzing the relationship between the source images and the second part is inferring the predicted image from the target image using the same relationship. For example, in Figure 1, the image a, the filtered image a' of a are source images, and image b is the target image of the analogy. The image b' can be inferred from b through the analyzed relationship between a and a' which converts RGB to grayscale.

Figure 1. Image analogy

In this paper, we will try to solve the image analogy problem. There are several attempts to deal with image analogy by leveraging various machine learning models. Among them, Reed et al. [1] proposed leveraging a DNN to perform an image analogy to learn the relationship between images and transform other images based on the relationship.

However, DNN-based models require a very large amount of data. In the case of image analogy, building such a dataset is difficult because image relationships can exist in countless ways such as blurring or texture transferring. And in order to learn one relationship, it is necessary to collect tens of thousands of image pairs (a:a') having the same relationship.

In order to solve these problem, we tried to use a Gaussian process (GP) instead of DNN to capture image relationships. The GP model is widely used for regression and classification. GP is nonparametric, meaning that the complexity of the model grows as more data points are received. The cost of THE training is $O(N^3)$ complexity, where N is the number of training data points. Therefore, as the amount of data increases, it becomes hard to use GP from the perspective of training time and memory cost. For the solution of this problem, we propose an efficient sampling method using the characteristics of GP that provide a variance value for new data.

II. Related Work

In this section, we survey related works that deal with image analogy and image generation with GP.

Reed et al [1] proposed a DNN that performs a visual analogy. They propose an end-to-end model that learns a mapping function which transforms images to a fixed vector space via neural embedding, which makes analogical reasoning simple. In experiments, they show the model works well in visual analogy problems on three datasets: 2D shapes, video game character sprites and CAD 3D car models. However, it is difficult to interpret the meaning of the hidden layers in their model. In addition many prior levels are still required to generate an accurate result.

Ashish Kapoor et al. [2] proposed GP for object categorization. They described object categorization as a fundamental problem in image understanding and processing, and it remains a challenging learning task. To solve these challenges, they introduced a new GP regression method using a local feature correspondence kernel and performed experiments to prove their method. The results of the experiments demonstrated good image categorization accuracy when compared to other methods, however, they did not

2375-9356/18/$31.00 ©2018 IEEE
DOI 10.1109/BigComp.2018.00086

provide conclusive proof that classification using a prior GP is inherently superior to other classification techniques.

III. METHOD

A. Network Topology

Figure 2 shows our proposed model. The model is composed of three parts, an encoder network, an increment function, and a decoder network. The encoder and decoder network used a convolutional neural network (CNN), and these perform the functions of converting an image to a vector and converting a vector to an image. The encoder and decoder networks are similar to the previously proposed model on Reed et al. [1]. However, the increment function changed to GP. The increment function using GP provides the probabilistic interpretation, this means that we can analyze the result of the model and utilize it to refit the model. And we redefine a formula for our GP model as follows:

$$\overline{b'} = D(E(b) + GP(E(a') - E(a), E(b))) \tag{1}$$

where $D(x)$ is the decoded image of vector x, $E(x)$ is the encoded vector of image x, GP is the GP increment function, and $\overline{b'}$ is the predicted output.

However, there is a problem in training the model. The encoder and decoder network using CNN are parametric, whereas GP is non-parametric. In order to handle this problem, the GP and encoder-decoder pair are trained separately. The encoder-decoder pair is trained as an autoencoder model that has the same number of input and output units and generates the same output image as the input image [3]. The decoder works in the opposite manner of the encoder in that it converts the vector into original image which means the encoder and decoder of the autoencoder can be considered as inversely related. By using this, the output of GP can be easily induced as the following (2) from (1).

$$\begin{aligned} GP(E(a') - E(a), E(b)) &= D^{-1}(b') - E(b) \\ &= E(b') - E(b) \end{aligned} \tag{2}$$

Therefore, to train the GP increment function, encoded a image, a' image and b image are used as input of GP increment function and encoded b' image subtracted by encoded b image is used as output of GP increment function.

B. Highest Variance Sampling

The attractive feature of GP is that it provides the variance of the new data points [4]. There is a high variance in regions where the model has a high uncertainty and the uncertainty is derived from a lack of training data for that region. In other words, if the variance of a new data point is determined to be high, it means that the new data point is different from the data already trained by the model. Therefore, selecting data with a high variance rather than data with a low variance can be helpful to ensure a well-distributed sampling.

Algorithm 1 Simple Highest Variance Sampling

Require: Training data: X
Number of data to be sampled in training data: N
Ensure: Trained $model$ with sampled data
1: x = n randomly sampled training data in X
2: $remainData$ = $X - x$
3: K = $N - n$
4: $model$ = train GP model with x
5: $varianceList$ = calculate variance of each data in $remainData$ using $model$
6: $sampledData$ = select K highest variance data in $varianceList$
7: $model$ = Additionally train $model$ with $sampledData$
8: **return** $model$

Algorithm 1 is a pseudo code of simple highest variance sampling (SHVS). First, n data are randomly sampled from the entire training data. The GP model is then trained with the sampled data and the variance for the remaining training data is calculated using the model. Then, the K highest variance data is extracted from the variance list and trained in the model where K means the remaining number to sample. As mentioned before, the K highest variance data was chosen because it is not the most similar to the previously trained data x. It can be also considered as selecting the farthest distance data from the trained data x. However, it does not guarantee that the K extracted data are not similar to each other and it even has a tendency to choose similar data which would result in a biased sampling result. If the variance of one piece of data is high, data similar to it will also have a high variance. Thus, similar data near a data point with a high variance will also be sampled together and therefore, it becomes difficult for the model to learn well-distributed data.

So as another option, the distance between instances can be used to select well-distributed data. And the correlation coefficients are often used to penalize the similarity between instances [5].

$$d_{ij} = \frac{1}{|\rho_{ij}|} \quad (where - 1 \leq \rho_{ij} \leq 1) \tag{3}$$

The distance between instance i and j is defined as inverse of absolute value of correlation coefficient as shown in (3). The correlation coefficient can be derived from the covariance matrix obtained by the GP model as shown in (4).

$$\rho_{ij} = \frac{cov_{ij}}{\sigma_i \sigma_j} \tag{4}$$

By the definition of distance, the distance between i and j increases as the absolute value of ρ_{ij} approaches 0 which means i and j are independent and the distance between i,j decreases as the absolute value of ρ_{ij} approaches 1.

Algorithm 2 is a pseudo code of far distance sampling (FDS). FDS is a method to perform a well-distributed

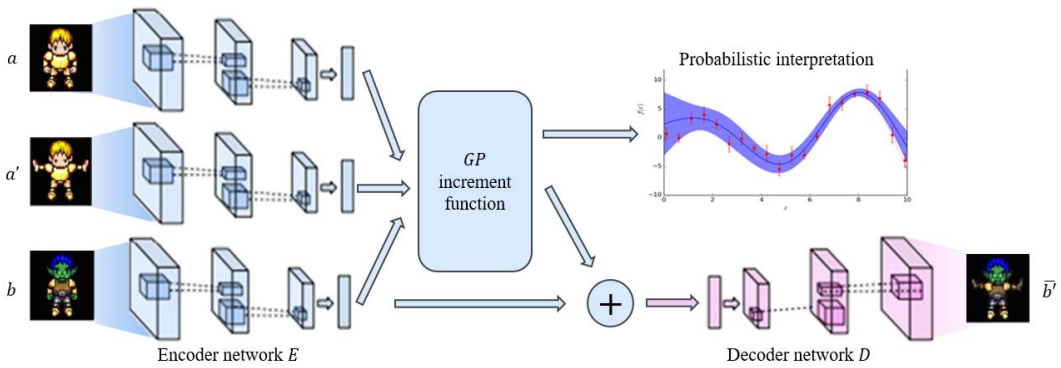

Figure 2. Proposed novel model architecture

Algorithm 2 Far Distance Sampling

Require: Training data: X
Number of data to be sampled in training data: N
Ensure: Trained $model$ with sampled data
1: $x = n$ randomly sampled training data in X
2: $remainData = X - x$
3: $K = N - n$
4: $model$ = train GP model with x
5: $varianceList$ = calculate variance of each data in $remainData$ using $model$
6: Z = select K' highest variance data in $varianceList$ (where $K' > K$)
7: $S = \max_{\forall s \in Z} \sum_{i,j \in s} d_{ij}$ (where s is the K size subset of Z)
8: $model$ = Additionally train $model$ with S
9: **return** $model$

sampling using distance. It is identical to the fifth line of Algorithm1. But FDS select the K' highest variance data from the variance list. After that, it finds a K size subset that maximizes the sum of the distances between the data. The distance between data is calculated by (3) using covariance matrix obtained by the GP model. By selecting the K' data which is larger than K, Z will more likely to include not only data points on high variance peak but also the top point of the low variance peak. And finding a K size subset that maximizes the sum of distances among Z leads to sampling data that is distributed on different peaks.

IV. Data

We use a dataset of animated 2D video game character sprites using graphics assets from the Liberated Pixel Cup[1] which are also used in the paper [1].

A. Experimental Setup

Typically, unlabeled data consists of samples that can be obtain relatively easily. However, since labeled data requires the augmentation of each unlabeled data with some sort of informative label, it is hard to obtain. Especially in the case of image analogy, it is necessary to collect image pairs (a:a') having the same relationship, which is too hard for a person to handle. In order to simulate this constraint, the autoencoder, which requires unsupervised learning, is trained with 10000 images out of a total of 20000 images. CNN-based autoencoder is used and the model is trained using the same image as input and output. The GP and DNN increment functions, which require supervised learning, were cross validated with 2000 training images and 8000 test images out of the remaining 10000 images.

B. Evaluation of Sampling Methods

Table I shows the difference in performance between SHVS, FDS and random sampling (RS) in the test data. In order to verify the performance of SHVS and FDS, the performances of RS with a different number of sample images are also calculated. The lower the sum of standard deviations, the better the sampling method is. Both methods proposed in this paper outperformed the random sampling method under the same conditions. GP with 1000 images sampled by SHVS is better than GP with 1000 images by RS but not better than 1100 images by RS. When 1000 images are sampled by FDS, the result is much better than 1200 images are sampled by RS.

Table I
Comparison between SHVS, FDS and RS on test data

Sampling method	Sum of standard deviation
Train with 1000 sample images by SHVS	393.734
Train with 1000 sample images by FDS	356.282
Train with 1000 sample images by RS	412.168
Train with 1100 sample images by RS	388.137
Train with 1200 sample images by RS	369.374
Train with 1300 sample images by RS	350.352
Train with 1400 sample images by RS	337.206

[1]http://lpc.opengameart.org/

C. Evaluation of GP Image Analogy

As mentioned in the experimental setup, The GP and DNN increment functions were cross validated with 2000 encoded training images and 8000 encoded test images. To compare the image analogy results of each GP and DNN, the GP increment function is trained with 1000 sampled images out of 2000 training images, and the DNN increment function is trained with all 2000 training images. Both the GP and DNN increment functions use the same trained encoder and decoder.

Figure 3 provides a qualitative comparison of our proposed model. Several samples are randomly chosen from test dataset and the two columns on the right are the predicted images of the GP and DNN for the target images b. The result shows that our image analogy method using the Gaussian process performs significantly better than DNN in qualitative perspectives. GP generate a well-predicted image that reflect the relation of a and a', whereas DNN cannot generate the proper predictive image due to a lack of data.

Table II provides a quantitative comparison of our proposed model. It shows the mean-squared pixel error of predicted image $\overline{b'}$ compared to the correct answer b' for the test dataset. The result demonstrates the large difference in performance between GP and DNN. The mean-squared pixel error of DNN is 100.599, which is about 770% higher than that of GP using RS. Also, we observe that FDS results in a significant improvement compared to RS. The mean-squared pixel error of FDS is 11.877, 9.2% lower than RS. However, SHVS did not significantly improve performance over RS. The mean-squared pixel error of SHVS is 12.768, only 2.5% lower than RS.

V. Conclusion

In this study, we introduce a novel image analogy method which adopts the Gaussian process. The Gaussian process regression as well as autoencoder are carefully designed in order to apply both algorithms to image analogy. In order to accelerate the training speed of the Gaussian process, we propose salient instance sampling as well. Two approaches

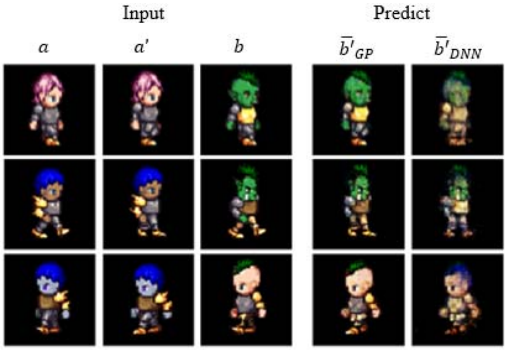

Figure 3. Comparison between predicted images of GP and DNN

Table II
Mean-squared pixel error on test data

Increment function	Mean-squared pixel error
DNN	100.599
GP using RS	13.067
GP using SHVS	12.768
GP using FDS	11.877

are adopted for salient instance sampling, which are simple highest variance sampling, partitioned highest variance sampling and far distance sampling. Our experiments shows that our proposed sampling methods outperform the random sampling method. Furthermore, the proposed image analogy method using the Gaussian process with salient instance sampling performs significantly better than DNN with a small dataset. In future work, we will verify the effectiveness of our methods using more complex datasets. Also, we will investigate different kernel functions for Gaussian process regression and different distance measurements for instance sampling in order to determine the best configuration for our proposed methods.

Acknowledgment

This work was supported by Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korean Government (MSIP) (No.2013-0-00131, Development of Knowledge Evolutionary WiseQA Platform Technology for Human Knowledge Augmented Services; and No.2017-0-00868, Development of Conversational Solution for Intelligent Chat Services Based on Pragmatic and Context Analysis of Dialogues).

References

[1] S. E. Reed, Y. Zhang, Y. Zhang, and H. Lee, "Deep visual analogy-making," in *Advances in Neural Information Processing Systems*, 2015, pp. 1252–1260.

[2] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell, "Gaussian processes for object categorization," *International journal of computer vision*, vol. 88, no. 2, pp. 169–188, 2010.

[3] C. Hong, J. Yu, J. Wan, D. Tao, and M. Wang, "Multimodal deep autoencoder for human pose recovery," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5659–5670, 2015.

[4] C. K. Williams and C. E. Rasmussen, "Gaussian processes for regression," in *Advances in neural information processing systems*, 1996, pp. 514–520.

[5] F. L. Minku, H. Inoue, and X. Yao, "Negative correlation in incremental learning," *Natural Computing*, vol. 8, no. 2, pp. 289–320, 2009.