GAN Data Augmentation Through Active Learning Inspired Sample Acquisition

Christopher Nielsen
University of Calgary
csnielse@ucalgary.ca

Michal Okoniewski
University of Calgary
okoniews@ucalgary.ca

Abstract

Data augmentation is frequently used to increase the effective training set size when training deep neural networks for supervised learning tasks. This technique is particularly beneficial when the size of the training set is small. Recently, data augmentation using GAN generated samples has been shown to provide performance improvement for supervised learning tasks. In this paper we propose a method of GAN data augmentation for image classification that uses the prediction uncertainty of the classifier network to determine the optimal GAN samples to augment the training set. We apply the acquisition function framework originally developed for active learning to evaluate the sample uncertainty. Preliminary experimental results are provided to demonstrate the benefit of this technique.

1. Introduction

Training deep neural networks for image classification typically requires a significant amount of labelled data. As datasets for specialized domains such as medical radiology are generally small and the associated labelling cost is large, data augmentation is commonly applied to increase the dataset size. Recently, strategies for data augmentation using a generative adversarial network (GAN) have been proposed and shown to achieve moderate success for medical image segmentation [1]. However, a question remains as to determining the optimal way to sample from the GAN latent space to form the augmented training set.

A closely related domain of research is known as active learning [2]. The purpose of active learning is to maximize classification performance while minimizing the amount of required labelled data. In active learning, a model is initially trained on a small dataset, then using what is known as an acquisition function, the prediction uncertainty for each of the unlabeled samples is evaluated. The samples with the greatest prediction uncertainty are then labelled by an external oracle, added to the training set, and the classification network is retrained. This process repeats in an iterative fashion until the desired performance is achieved.

The purpose of this work is to propose a data augmentation strategy that uses the acquisition functions developed for active learning to choose samples generated by a label conditioned GAN to augment the training set. This functionality is used to develop a classification system that iterates between training the classifier and expanding the training set by selecting GAN samples on the basis of the classifier prediction uncertainty. Experimental analysis will examine the performance of this technique for classification using the MNIST dataset under GAN architectures of varying capacity.

2. Related Work

Suppose that we have a dataset consisting of input data $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$ with a corresponding set of labels $\mathbf{Y} = \{y_1, y_2, \cdots, y_N\}$ and we have trained a neural network classifier to estimate the discriminative distribution $P(y \mid \mathbf{x}, \mathbf{X}, \mathbf{Y})$ such that we can make inferences on this distribution to find the optimal label to assign to a new data point $\mathbf{x}$. Given that the classifier is defined by parameters $\boldsymbol{\omega}$, we can expand the discriminative distribution using Bayes' rule as follows

$$\begin{aligned} & P(y \mid \mathbf{x}, \mathbf{X}, \mathbf{Y}) \\ &= \int_{\omega} P(y, \boldsymbol{\omega} \mid \mathbf{x}, \mathbf{X}, \mathbf{Y})\, d\boldsymbol{\omega} \\ &= \int_{\omega} P(y \mid \mathbf{x}, \mathbf{X}, \mathbf{Y}, \boldsymbol{\omega})\, P(\boldsymbol{\omega} \mid \mathbf{x}, \mathbf{X}, \mathbf{Y})\, d\boldsymbol{\omega} \end{aligned} \tag{1.1}$$

A pertinent question is how we can assess the uncertainty that the classifier has about the estimates it makes so that we can assign a confidence level to the assigned classification label. Kendall et al. [3] and Gal et al. [4] show how dropout can be used as technique of sampling from a distribution which approximates $P(\boldsymbol{\omega} \mid \mathbf{x}, \mathbf{X}, \mathbf{Y})$ by assuming Bernoulli prior distributions for the weights. Dropout is a technique which was proposed originally to regularize a neural network for prevention of overfitting [6]. The basic premise is that during training, a Bernoulli random variable is sampled for each network parameter where dropout is used. This sampled value acts as a multiplicative mask for the parameter. In other words, when the sampled value is 1, the parameter keeps its value, otherwise the parameter is

dropped for the training iteration (assigned a value of 0). The motivation behind this technique is to stochastically create subnetworks within the larger network, such that the network must learn redundancy which combats overfitting. For Bayesian uncertainty analysis, when we use dropout, we can consider each parameter to be sampled from a scaled Bernoulli distribution. Kendall et al. [3] show how by using this formulation, we can develop a Monte Carlo method using dropout for sampling from the desired distribution $P(\boldsymbol{\omega} \mid \mathbf{x}, \mathbf{X}, \mathbf{Y})$. The final calculation is given as follows

$$P(y \mid \mathbf{x}, \mathbf{X}, \mathbf{Y}) \approx \frac{1}{N} \sum_{n=1}^{N} P(y \mid \hat{\boldsymbol{\omega}}_n, \mathbf{x}, \mathbf{X}, \mathbf{Y}) \quad (1.2)$$

Where $\hat{\boldsymbol{\omega}}_n$ are the parameters of the network sampled in the nth Monte Carlo dropout sample. We will refer to this sampling technique as MC dropout. We can use this approximation of the predictive distribution to better estimate the network uncertainty. To rank samples by their uncertainty we use a scoring metric called an acquisition function. The Bayesian Active Learning by Disagreement (BALD) acquisition function proposed by Houlsby et al. [8] is defined as follows

$$\begin{aligned} U(\mathbf{x}) = & H\left[P(y \mid \mathbf{x}, \mathbf{X}, \mathbf{Y})\right] \\ & - \mathbb{E}_{P(\boldsymbol{\omega} \mid \mathbf{X}, \mathbf{Y})}\left[H\left[P(y \mid \mathbf{x}, \boldsymbol{\omega})\right]\right] \end{aligned} \quad (1.3)$$

Computationally, this can be approximated using the MC dropout samples as

$$\begin{aligned} U(\mathbf{x}) \approx & H\left[\frac{1}{N} \sum_{n=1}^{N} P(y \mid \mathbf{x}, \boldsymbol{\omega}_n)\right] \\ & - \frac{1}{N} \sum_{n=1}^{N} H\left[P(y \mid \mathbf{x}, \boldsymbol{\omega}_n)\right] \end{aligned} \quad (1.4)$$

where N is the number of MC samples, and $\boldsymbol{\omega}_n$ are the parameters of the network sampled for the nth MC dropout sample. Data points with high entropy for the average predictive distribution of the MC dropout samples, but low average entropy for the entropy of each of the sampled predictive distribution will have a high BALD score indicating that the network is uncertainty about the prediction. The intuition behind this metric is that if the dropout sampling of the weights causes the network to change its prediction, then the network is considered uncertain about the sample prediction.

The initial paper on GANs was written by Goodfellow et al. [10] and the focus of this original work was to describe the minimax competition between the discriminator and generator networks. An extension was made by Mirza et al. [11] to condition both the generator and discriminator models on the label of the training data, enabling samples to be generated from specific class labels. This work was expanded by Radford et al. [12] when the DCGAN architecture was developed which used deep convolutional neural networks for both the generator and discriminator models. Additionally, it was shown how generated samples from the trained GANs could be used for semi-supervised learning, where the initial layers of the discriminator are used as a feature extractor to train a classification model. Due to the instability of training GANs using original loss function presented by Goodfellow et al. [10], Gulrajani et al. [13] released an improved loss function based on the Wasserstein distance. Theoretically the Wasserstein loss has smoother gradients and greater stability over the loss function proposed by [10]. The PGGAN architecture was released in 2017 and provided an approach to train a GAN architecture by training the discriminator and generator models on lower resolution samples before progressively growing toward high resolution samples [9].

3. System Overview

Three different GAN models were used for the experimental work in this paper, denoted as Small-DCGAN, Large-DCGAN, and PGGAN with increasing capacity respectively. The Small-DCGAN and Large-DCGAN are variants of the original DCGAN model proposed by [12]. The PGGAN model closely resembles the model published by [9]. The convolutional neural network (CNN) architecture for the classifier model was a simple five layer network similar to that used by [14].

The algorithm for the processing performed during each iteration of the classification training loop is shown in Figure 3-1. We start iteration step N in possession of the current trained classifier network and the current training set. To perform an iteration of the training loop, samples from the data source are used to compute the classifier network posterior estimates through MC dropout. Next, an acquisition function is used to process the posterior estimates and assign each image sample a score. The samples with the highest scores are added to the training set for iteration step $N+1$ and used to train the resulting classifier for iteration step $N+1$. This process repeats until desired convergence is met or the predefined number of iterations are completed. For the base case when $N = 0$, the classifier network is initialized with random parameter values. The acquisition functions used for this paper are random sampling and BALD. Random sampling simply involves selecting random images from the data source to become part of the training set for the next iteration. BALD acquisition involves computing the score described in equation (1.4). After the scores are computed, they are sorted and the images with the highest scores are sampled and added to the training set for the next iteration. The overall procedure for this system is described in Algorithm 1.

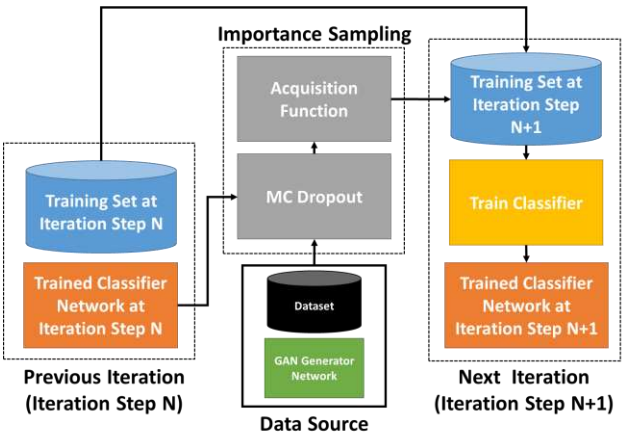

Figure 1: Overview of the classification training loop.

Algorithm 1: Classifier training using GAN data augmentation. Samples selected from latent space using Bayesian acquisition.

Data: Labelled dataset D_{raw}, Number of samples to append each iteration N_{iter}

```
Train GAN using D_raw
Generate sample set D_generated from GAN
Form augmented dataset D_augmented ← {D_raw, D_generated}
Set D_training ← {}
Initialize CNN parameters
for Number of CNN training iterations do
    for x_n ∈ D_augmented do
        Compute predicted CNN probabilities φ using MC-Dropout
        Evaluate acquisition function U(φ)
    end
    Append the samples x_n with the N_iter largest acquisition scores to D_training
    Retrain CNN using D_training
    Evaluate CNN balanced accuracy using test set
end
```

4. Experimental Procedure

At the start of each iteration, the classification network was initialized with a set of new random weights. During each iteration the chosen acquisition function was used to sample 10 images from the given data source. These new images were added to the training dataset. The classification network was then trained using this dataset for 100 epochs. The resulting final balanced accuracy was then computed and stored for each iteration. This process continued for 50 iterations. At this point the training set size had reached 500 images. Each experiment examined the performance when the GAN data sources were used to augment the raw dataset i.e. (the classifier was trained on a dataset consisting of the raw MNIST dataset combined with data from one of the three possible GAN data sources: Small-DCGAN, Large-DCGAN, and PGGAN). For all experiments, classification performance was measured using each combination of data source and acquisition function. Additionally, each experiment was repeated 4 times to establish a confidence interval for the accuracy estimate. All experiments were performed using the Keras library [7].

In Figure 2, the results of training the classifiers under BALD acquisition are shown together with the performance of the raw data classifier trained under random sampling added as a baseline. We see that the performance of the classifiers trained using augmentation by BALD acquisition outperformed the classifiers trained on the raw datasets. From the final test accuracies, we observed that the BALD PGGAN augmented dataset had a balanced accuracy increase of 3.82 percentage points over the Random Raw dataset, and an increase of 0.86 percentage points over the BALD Raw dataset. This indicates the potential advantage of the proposed method.

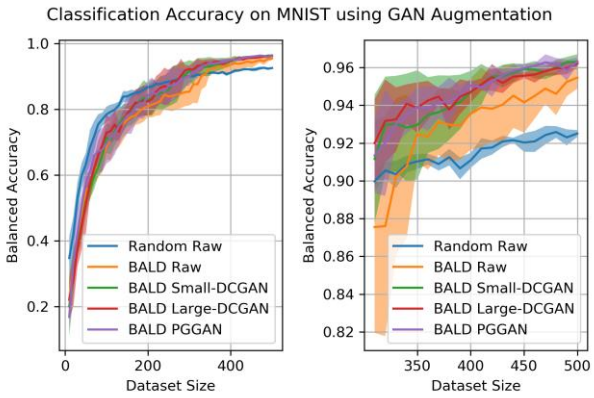

Figure 2: Plot of the classifiers trained using GAN augmented MNIST data. The plot on the left shows the balanced accuracy for all training iterations, while the plot on the right shows the balanced accuracy for the final 20 iterations. The shaded area around each line signifies a confidence interval of one standard deviation.

5. Conclusion

Through the preliminary experimental results, we have seen that training a classification network using a dataset augmented with synthetic GAN samples can improve the overall performance of the classifier. Additionally, the acquisition function sampling mechanism was shown to further improve the classifier performance, especially for the GANs with lower capacity. In conclusion, the preliminary experimental results provided in this paper demonstrate that GAN augmentation using Bayesian uncertainty analysis is advantageous for image classification.

References

[1] C. Bowles, L. Chen, R. Guerrero, P. Bentley, R. Gunn, A. Hammers, D. A. Dickie, M. V. Hernndez, J. Wardlaw, and D. Rueckert, "GAN augmentation: Augmenting training data using generative adversarial networks," *arXiv preprint arXiv:1810.10863*, 2018.

[2] B. Settles, "From theories to queries: Active learning in practice," *Active Learning and Experimental Design Workshop, vol 16, JMLR Proceedings, Sardinia, pp 1–18*, 2010.

[3] A. Kendall, Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" *ArXiv:1703.04977*, 2017.

[4] Y. Gal, Z. Ghahramani, "Dropout as a bayesian approximation: representing model uncertainty in deep learning," *arXiv preprint arXiv:1506.02142*, 2015.

[5] Y. Gal, "Uncertainty in Deep Learning," *PhD thesis, University of Cambridge*, 2016.

[6] N. Srivastava, G Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res., 15(1):1929–1958*, 2014.

[7] F. Chollet, et al. "Keras," *https://keras.io*, 2015.

[8] N. Houlsby, F. Huszar, Z. Ghahramani, and M. Lengyel, "Bayesian active learning for classification and preference learning," *arXiv preprint arXiv:1112.5745*, 2011.

[9] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, et al. "Generative adversarial nets," *NIPS*, 2014.

[11] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *CoRR, abs/1411.1784*, 2014.

[12] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[13] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," *CoRR, abs/1704.00028*, 2017.

[14] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. *arXiv preprint arXiv:1703.02910*, 2017.