Generating Computed Tomography Brain Scans with a DCGAN

Christopher Meszaros, Federica Comuni

***Abstract*—Generative Adversarial Networks (GAN) is a recently introduced deep learning approach to generating new samples by learning the training data's distribution. Applications are extensive and range from generating images (faces, deep fakes etc..) to photo blending, change of image scenery or subject's pose, data augmentation, and others. In our project, we focus on generating a certain type of medical imaging known as brain computed tomography scan. The produced images show that the GAN effectively learned some of the most salient features, e.g. the skull's oval shape, and are considerably similar to the real CT scans, albeit not enough to be mistaken for real by a specialist. A conditional GAN was also implemented to produce brain scans according to a given label, in this case healthy or hemorrhagic. The samples generated by the conditional GAN were tested on a classifier trained on real data, which classified the fake samples with 45.3% accuracy.**

Introduction

COMPUTED tomography (CT) brain scans are medical images used to detect brain hemorrhages and other pathological conditions of the cranium. Classifications models can be built in order to detect different types of hemorrhages, e.g. subdural or intraparenchymal, using annotated CT brain scans. Good performing models often need to train on a great amount of data in order to reduce variance and overfitting. However, acquiring new CT brain scans is costly because it requires domain knowledge from a neurologist or radiologist to annotate a CT scan. The annotation procedure can also be time consuming if it entails image segmentation or is performed on three-dimensional image data. The CT scan is either annotated as healthy or as having a certain type of brain hemorrhage; the datasets are usually imbalanced, as the amount of CT scans that are annotated as hemorrhagic might be low compared to CT scans with a healthy brain.

The mentioned issues can be solved by using GANs for data augmentation purposes. The idea is to train a generative model which learns the data's distribution; in the case of image data, the aim is to learn the distribution of the pixels' color values in the images. Once this distribution is learned, it is then possible to produce newly generated samples that can be used to train the classifier.

In this project we have implemented a GAN and trained it on a subset of the dataset for intracranial hemorrhage detection released by the Radiological Society of North America and available on Kaggle [1]. The GAN's architecture has been structured to produce 128x128 images. The generated images were then proposed to a specialist in neurointensive care and to a medical student to be distinguished between real and fake samples. Since the dataset includes samples from different classes (healthy and hemorrhagic), a conditional GAN was implemented to generate samples from a specific class. The samples produced by the conditional GAN were then evaluated by a binary classifier previously trained on real samples.

Background and related work

The aforementioned issues regarding cost inefficiency and requirement for expertise of medical datasets annotation tasks and the tendency of medical data to be scarce, expensive to produce or imbalanced, are well-known in medical settings [2]. For this reason, data augmentation and medical image synthesis with GAN have risen in popularity in the past few years. In 2018, Shin et al. [2] and Bowles et al. [3] performed brain segmentation after generating MRI samples with a Deep Convolutional GAN (DCGAN) and a Progressive Growing of GANs (PGGAN), respectively. Also in 2018, Kazuhiro et al. [4] proposed real and GAN-generated MRI samples to five blinded radiologists and assessed their accuracy in distinguishing the two sets. The radiologists mistook 34% of the fake images for real samples, on average. GANs have

therefore become established in the counteraction of possible issues in healthcare data quality, although we have not found any previous studies concerning synthesis of CT brain scans.

What are GANs?

GANs were first introduced in a paper 2014 [5]. The authors describe the use of two models: *Discriminator* (*D*) and *Generator* (*G*) having a conflict of interest with respect to their optimization (hence the term *adversarial*). The discriminator's objective is to distinguish between the real and the fake (produced by generator) samples, while the generator's objective is to generate good enough images as to fool the discriminator.

Let $\mathbf{z} \sim p_z(\mathbf{z})$ a noise vector and $\mathbf{x} \sim p_x(\mathbf{x})$ be a data sample generated from the training data's true distribution. The authors proposed the following min-max [1] set up between the function *G* and *D*. ($\min_G \max_D$)

$$E_{\mathbf{x}\sim p_{data}(\mathbf{x})}[log(D(\mathbf{x}))]+ \\ E_{\mathbf{z}\sim p_z(\mathbf{z})}[log(1-D(G(\mathbf{z})))] \quad (1)$$

The *discriminator* trying maximize the probability of assigning the correct label to both training examples and samples from G (distinguish between real or fake) using both terms above. The *Generator* uses the second term minimize errors made by the generated pictures (i.e., generating good pictures to fool discriminator).

Since $G(z)$ and $D(x)$ are differentiable functions, the networks can be trained in backpropagation fashion. The idea is to take the noise vector $\mathbf{z}$ and transform it according to the training data distribution. Initially, the generator is producing white noise which the discriminator can easily distinguish. This makes it hard for the *G* to learn well in early training for Equation 1. This is because $log(1 - D(G(\mathbf{z})))$ saturates since D can easily distinguish G. So the authors suggested training G to maximize $\max_G log(D(G(\mathbf{z})))$ instead to speed up training for the generator in early iterations.

[1] MinMax can be used in two-player games that decide what action to take. It can be described as trying to minimize the loss for the worst possible case. The worst case refers to the opponents players trying to make winning moves. In GANs the two players are G and D

An extension to the GANs is the deep convolutional generative adversarial networks (DCGANs). This model was first introduced in 2015 by Radford and Metz [6] and consists of implementing both generator and discriminator as convolutional neural networks. The generator takes a 100 dimensional vector of random noise following a normal distribution as input and gradually converts it to a picture of desired dimensions (in this case, 128x128x1) by gradually decreasing the number of filters per layer and increasing the feature map size of the convolutional transpose layers. The discriminator is implemented as a binary classifier receiving images as input and passing them through convolutional layers increasing the number of filters per layer and decreasing the feature map size until the 1x1x1 output is passed through the sigmoid function. Radford et al. stress the importance of performing spatial up- and down-sampling through strided convolutions instead of pooling layers, to avoid the use of fully connected layers in the discriminator, and to use batch normalization in both networks to "*stabilize learning and help gradient flow*" [6].

Another extension to the GAN is the conditional GAN (cGAN). Given data which contains several classes, the original GAN approach has no way of controlling from which class the sample is generated. The purpose of the cGAN is therefore to condition the GAN from which class the G should sample from [7]. Equation 1 can be reformulated to incroporate the conditonal for both discriminator $D(\mathbf{x}|\mathbf{y})$ and the generator $G(\mathbf{z}|\mathbf{y})$ where $\mathbf{y}$ represents the one-hot encoding of class labels

$$E_{\mathbf{x}\sim p_{data}(\mathbf{x})}[log(D(\mathbf{x}|\mathbf{y}))]+ \\ E_{\mathbf{z}\sim p_z(\mathbf{z})}[log(1-D(G(\mathbf{z}|\mathbf{y})))] \quad (2)$$

METHOD

The brain scans from the RSNA dataset are medical images annotated with metadata in DICOM format; conversion to PNG format with the PyDicom library [8] was therefore required for training the models. Since the pixel array of each image spanned over a 0-2550 scale, each image was rescaled to a 0-255 range as explained in [9]. The images were then loaded through a `PyTorch ↪ ImageFolder` and used following resizing, grey scaling, conversion to tensors and normalization to mean of 0.5 and standard deviation of 0.5. The pictures were grey scaled because, in the

case of CT scans, color was not deemed necessary by us to represent the most salient features. For our implementation of the DCGAN, we made use and re-adapted the DCGAN tutoral from PyTorch website [10]. The tutorial strictly attains to the model architecture and hyperparameter recommendations expressed by Radford et al., e.g. it uses a learning rate of 0.0002 and a $\beta 1$ value of 0.5 for the Adam optimizer to stabilize training, the ReLU activation function for the generator and LeakyReLU for the discriminator [6]. The tutorial is suited for producing 64x64 images; since the original dataset consists of 512x512 images, we first re-adapted the network architecture to generate images of that size and then 256x256, but in both cases we incurred in out-of-memory problems and therefore finally opted for 128x128 images. At first the generator and discriminator failed to converge, with the discriminator reaching excellent accuracy in just one epoch and the generator failing to produce anything but noise. To counteract this problem and prevent the discriminator from becoming too good, too fast, we reduced the discriminator's starting number of filters to be a quarter of the generator's. Also, to counteract some of the typical points of failure for GANs such as convergence failure or model collapse, we integrated the model with the possibility to use Wasserstein loss instead of mini-max loss, which aims to prevent collapsing [11], and with the option to add random noise to the discriminator's input. However, the results obtained with these modifications did not achieve the quality of the standard mini-max loss GAN, which is the model we have used to generate the pictures displayed in the Results section.

To implement the cGAN we reformulated the G and D architecture to include the class labels. So for each forward pass of a batch, we also send in the one-hot-encoding of the class labels for each batch. The visual explanation can be shown in Figure 1.

The labels in our data is located in a CSV which uses the filename. Therefore, we needed to use a modified version of `ImageDataFolder` class that also returns the filepath for each batch, so we can look up our labels in the CSV. We found an online solution for this [12]

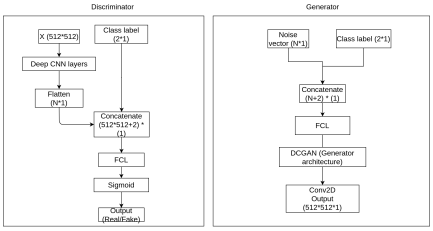

Figure 1. cGAN implementation

Results and Discussion

To assess the performance of GANs, both quantitative and qualitative evaluations can be carried out. Quantitative evaluations use computational methods to automatically produce a performance score, while qualitative methods use human judges to distinguish between real and fake. For a qualitative assessment we chose to evaluate the generated pictures thanks to a specialist in neuro-intensive care and a medical student. The specialist had domain expertise in analysing the CT scans, as well as experience with GAN-generated images, while the medical student had limited experience in both fields. We proposed them a grid of 24 pictures of which half were fake and half were real. The real samples were picked randomly while, for the fake samples, the most realistic were picked. While the specialist was able to detect the fake scans with 100% accuracy, the medical student did the same with 50% accuracy (on average, 25% of the fake samples were mistaken for real). Figure 2 shows the grid of samples proposed to the evaluators.

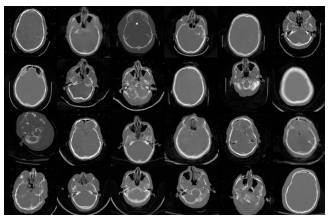

Figure 2. Array of fake and real scans proposed to the human evaluators

Our generated images suggest that the GAN got appreciably close to learning the distribution of the training data, by showing signs like the oval shape of the skull, the eye sockets, some facial bone features, and so on. However, the fake scans are overall quite blurry and present specks of white

in the background (probably "residues" of superimpositions of fair features throughout different samples) and lower definition than real scans.

Ideally, one should use quantitative measure since there are downsides to qualitative approaches: human bias, the requirement of domain knowledge, and the possibility to evaluate only a subset of images. We therefore considered as a quantitative measure the mean output of the discriminator on the batch of fake images throughout the last 20 epochs of training, which is 0.314; i.e., during the last 20 epochs, the discriminator was 31.4% confident that a fake picture was real, on average. This confidence is still far from the theoretical 50% proposed in the original paper by Goodfellow et al. [5]. Furthermore, this measure is sensitive to the "goodness" of the discriminator, as a poor discriminator will more easily mistake fake samples for real and therefore yield a higher confidence. The authors of the original GAN paper suggested average-log-likelihood.

An more popular quantitative approach is Inception score. This score uses a trained classifier to determine the cross-entropy loss for the specific class the generator was conditioned to generate [13]. This gives a good measures for how well the cGAN is able to produce a sample from a specific class.

We trained a binary classifier on real healthy and hemorrhagic samples. The train/val accuracy were 77% and 68% respectively. GAN was instructed to produce as FAKE hemorrhagic Scans, which the trained classifier detected with 45.3% accuracy. This measure is also susceptible to the effectiveness of the classifier in distinguishing the two classes; its performance, however, suggests that the cGAN did not captures features of typical of hemorrhagic scans.

Conclusion

The study explored the possibility of generating medical imaging samples through the use of a regular and a conditional GAN for data augmentation purposes. The generated samples showed that model effectively learned some of the most salient features of the brain scans. Scans evaluation by human subjects varied depending on the subject's expertise in medical scans and possibly familiarity with GANs. Computational evaluation showed that the models could be improved to generate more realistic and more class-representative images. Future work might therefore focus on such improvements through, for example, the implementation of a PGGAN, or the enlargement of the current model; the conditional GAN might also be extended to generate samples from the many types of intracranial hemorrhage.

References

[1] "Kaggle - rsna intracranial hemorrhage detection." [Online]. Available: https://www.kaggle.com/c/rsna-intracranial-hemorrhage-detection

[2] H. Shin, N. A. Tenenholtz, J. K. Rogers, C. G. Schwarz, M. L. Senjem, J. L. Gunter, K. P. Andriole, and M. Michalski, "Medical image synthesis for data augmentation and anonymization using generative adversarial networks," *CoRR*, vol. abs/1807.10225, 2018. [Online]. Available: http://arxiv.org/abs/1807.10225

[3] C. Bowles, L. Chen, R. Guerrero, P. Bentley, R. N. Gunn, A. Hammers, D. A. Dickie, M. del C. Valdés Hernández, J. M. Wardlaw, and D. Rueckert, "GAN augmentation: Augmenting training data using generative adversarial networks," *CoRR*, vol. abs/1810.10863, 2018. [Online]. Available: http://arxiv.org/abs/1810.10863

[4] P. Kazuhiro, R. Werner, F. Toriumi, M. Javadi, M. Pomper, L. Solnes, F. Verde, T. Higuchi, and S. Rowe, "Generative adversarial networks for the creation of realistic artificial brain magnetic resonance images," *Tomography*, vol. 4, no. 4, pp. 159–163, Dec. 2018. [Online]. Available: https://doi.org/10.18383/j.tom.2018.00042

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, and A. Courville, "Generative adversarial networks," *ArXiv, abs/1406.2661*, 2014.

[6] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2016.

[7] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *CoRR*, vol. abs/1411.1784, 2014. [Online]. Available: http://arxiv.org/abs/1411.1784

[8] "Pydicom documentation." [Online]. Available: https://pydicom.github.io/pydicom/stable/

[9] "Python convert .dcm to .png." [Online]. Available: https://stackoverflow.com/questions/60219622/python-convert-dcm-to-png-images-are-too-bright

[10] "Pytorch - dcgan tutorial." [Online]. Available: https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html

[11] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," 2017.

[12] "Github - pytorch image file paths with dataset dataloader." [Online]. Available: https://gist.github.com/andrewjong/6b02ff237533b3b2c554701fb53d5c4d

[13] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," 2016.