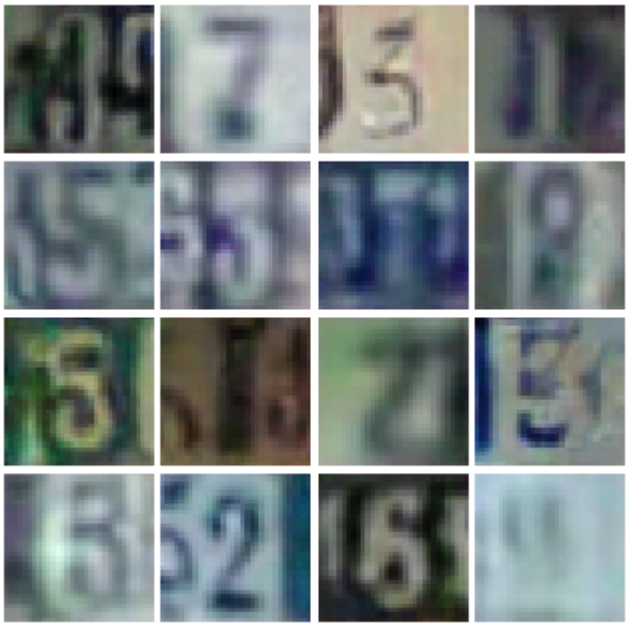
In [ ]:
```
# This output is your answer - please include in the final report
print("Vanilla GAN final image:")
fin = (images[-1] - images[-1].min()) / (images[-1].max() - images[-1].mi
show_images(fin.reshape(-1, 3, 32, 32))
plt.show()
```

```
Vanilla GAN final image:
```



Reflections (2 Points)

[TODO]: Reflect on the GAN Training Process. What were some of the difficulties? How good were the generated outputs?

After understanding the structure of how the training should happen, the difficulties were a matter of tuning (although I relied on the suggested hyperparameters in the notebook, the network's architecture was still flexible).

For my first try, I just followed the suggestions for both discriminator and generator, and after knowing that my code was running smooth I experimented with the addition of batchnorm layers (following hte guidelines of https://arxiv.org/pdf/1511.06434.pdf).

The outputs were not perfect, but not bad. In a qualitative look, some generated images are very similar to the ones from the training set, and could even fool me.

[TODO] What is a mode collapse in GAN training, can you find any examples in your training?

Mode collapse in GAN training is when the outputs of the generator is not reflective of the actual dataset in the sense of not being as diverse. The symptoms are not being able to generate all the available "labels", or only being able to output that is very close to the training set. It basically means that the generator was not able to learn the "meaning" of all of the data.

Yes. Despite not having quantitative results, I can see that the generator tends to output certain numbers more than others. For example, the numbers 4 and 9 are barely seen.

[TODO] Talk about the trends that you see in the losses for the generator and the discriminator. Explain why those values might make sense given the quality of the images generated after a given number of itterations.

I see a cyclic trend that the losses follow: in an iteration where the discriminator loss goes down, the generator loss goes up; when the generator loss goes down, the discriminator loss goes up. It makes sense because they are two competing networks. The way I see it, as the discriminator gets better, the generator has a harder time to fool it, and when it does, then the discriminator will fight against that.

Because of that it makes sense that the losses don't just go down as they usually do for common classification tasks, but instead they bounce around some number (in my case between 0.7 and 1.6).

Saving Models

In [ ]:
```
# You must save these models to run the next cells
torch.save(D.state_dict(), "d_smart.pth")
torch.save(G.state_dict(), "g_smart.pth")
```

Evaluating GANS

Evaluating via FID Score

It is hard to evaluate GANs effectiveness quantiatively. One way we can do this is by calculating FID (Frechet-Inception-Distance).

Website: https://wandb.ai/ayush-thakur/gan-evaluation/reports/How-to-Evaluate-GANs-using-Frechet-Inception-Distance-FID---Vmlldzo0MTAxOTI

But for this assignment, we will visually evaluate the GAN outputs.

TODO: Evaluating Discriminator

Not graded but for fun see how well your discriminator does against a sample generator.

In [ ]:
```
from part1_GANs.gan_pytorch import get_optimizer, eval_gan, eval_discrimi
from part1_GANs.gan_pytorch import generator, discriminator, discriminato

D = discriminator().type(dtype)
D.load_state_dict(torch.load("d_smart.pth"))
D.eval()

# oracle_g = torch.jit.load("../test_resources/g_smart_o.pt")
oracle_g = torch.jit.load("test_resources/g_smart_o.pt")
oracle_g.eval()

device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

accuracy, p, r = eval_discriminator(D, oracle_g, loader_val, device)

# Test accuracy is greater than 0.55
assert accuracy > 0.55
```

```
Accuracy: 0.6543386530748663
Precision: 0.9170792402673282
Recall: 0.3265662854289002
```

Preparing for Submission

Run the following cell to collect `hw4_submission_part1.zip`. You will submit this to HW4: Part 2 - GANs on gradescope. Make sure to also export a PDF of this jupyter notebook and attach that to the end of your theory section. This PDF must show your answers to all the questions in the document, please include the final photos that you generate as well. Do not include all of your training images! You will not be given credit for anything that is not visible to us in this PDF.

In [ ]:
```
!sh collect_submission_part1.sh
```

```
sh: 0: cannot open collect_submission_part1.sh: No such file
```

Contributers

- Manav Agrawal (Lead)
- Matthew Bronars
- Mihir Bafna