Date: 17th January 2023 Project supervisor: 韩劲松

Course: Computer Network Institute: 计算机科学与技术学院

浙江大学

语义通信的安全性探索
——基于白盒的对抗样本攻击

王晶晶 3200104880 信息安全

陈乐祺 3200106049 信息安全

曹颢玮 3200103998 信息安全

Contents

1 语义通信

随着无线通信智能化应用需求的快速提升，未来通信网络将从单纯追求高传输速率的传统架构向面向万物智联的全新架构转变。Weaver 定义了通信问题的三个层级为**语法**、**语义**、**语用**。语法通信确保消息或信号的无差错的传输。语义通信提取和理解信号中的含义，让接收端不仅是听到，而是听懂。语用通信让接收端理解语义后，执行所期望的特定任务。本次实验主要关注语义通信的层面。

语义通信最重要的特点就是可以通过探索语义信息来显着提高传输效率。通讯的初衷是要“达意”，即让接收者明白或理解发送者的本意. 通信真正的目的是接收方理解发送方的信息含义，降低接收者对信息的不确定性或者说使接收到的信息熵减少至 0，使接收者正确理解发送者的信息内容。是以，语义通信主要关注载波中的信息内容，不再关注载波波形的具体形式，而现有通讯恰恰相反，只关注载波形式而不关注内容。基于这样的目的，语义通信在一定程度上可以显著减小信息传输过程中的数据量，以此提高传输效率。

Zhang et al. (2022) 提出了一个基于深度学习的语义通信系统。其核心部分是一个语义编码网络 (semantic coding network)。为了实现任务不敏感的目标，也即语义通信系统能够实现在发送端不知道语用任务、仅接收端知道任务的语义通信，他们还引入了数据自适应网络 (data adaptation network)，以对任务进行风格迁移。

语义编码网络是在训练阶段基于经验数据库对译码器和编码器进行同步训练。整个语义编码网络的训练流程是在每轮迭代中，首先由接收端计算损失函数和梯度然后更新译码器，而编码器的更新首先需要接收端将信息反馈给发送端，发送端在计算过后再更新编码器网络。

数据自适应网络将可观测的数据迁移到训练语音编码网络时使用到的经验数据库，整个迁移过程只需要在发送端单独执行，不需要接收端参与，同时之前训练好的语义编码器和译码器也不需要重新训练。通过数据自适应网络，可以实现仅凭少量标定数据的训练，并且重复使用现有的语义编码网络，保证收发双方的隐私。

2 对抗样本攻击

对抗样本（Adversarial examples）是指为了发现某些不被人们注意的特征而使最大化模型的损失函数，从而在数据集中通过故意添加细微的干扰（往往是人类肉眼所无法察觉的干扰）所形成的输入样本，会导致模型以高置信度给出一个错误的输出。Figure 1是一个经典的对抗样本攻击，给熊猫的图片叠加上一个轻微的干扰之后，神经网络将其识别为长臂猿。

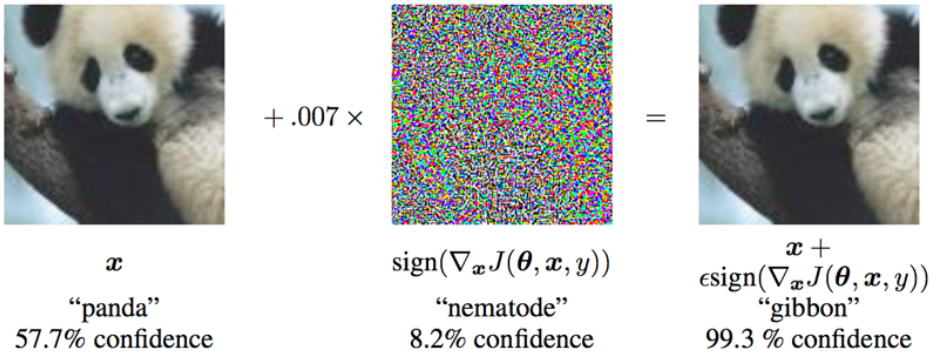


Figure 1: classical adversarial attack

一般来说，对抗样本攻击可以分为白盒攻击和黑盒攻击。

白盒攻击

白盒攻击假定攻击者能够获知机器学习所使用的算法，以及算法所使用的参数。攻击者在产生对抗性攻击数据的过程中能够与机器学习的系统有所交互，最主要的是知道经过训练的模型的所有参数。由于梯度等信息的可知性，这种攻击相对更加容易实现。

最经典的两种白盒对抗样本攻击是由Goodfellow et al. (2014) 提出的 FGSM(Fast Gradient Sign Method) 和由Madry et al. (2017) 提出的 PGD(Projected Gradient Descent)。本次实验主要关注这两种攻击对于语义通信系统的效果。

FGSM 在白盒环境下，FGSM 通过计算损失对于图像的梯度，然后用符号函数得到其具体的梯度方向，接着乘以一个控制扰动大小的参数 ϵ，让图片向梯度上升的方向添加噪声，直到模型分类错误。FGSM 得到的样本如下公式所述。

$$x' = x + \epsilon \cdot sign(\nabla_x L(\theta, x, y))$$

PGD PGD 攻击是一种迭代攻击，可以看作是 FGSM 的翻版——K-FGSM (K 表示迭代的次数)。FGSM 是仅仅做一次迭代，走一大步，而 PGD 是做多次迭代，每次走一小步，每次迭代都会将扰动 clip 到规定范围内。FGSM 直接通过 epsilon 参数一下子算出了对抗扰动，这样得到的可能不是最优的。因此 PGD 进行了改进，多迭代几次，慢慢找到最优的扰动。PGD 得到的样本如下公式所述。

$$x_{t+1} = \Pi_{x+S}(x_t + \alpha \cdot sign(\nabla_x L(x_t, y))$$

黑盒攻击

白盒攻击需要获取神经网络完整的信息，这在现实中不一定能够实现，相对更加常见的是黑盒攻击：攻击者并不知道机器学习所使用的算法和参数，但攻击者仍能与机器学习的系统有所交互，一般攻击者被赋予查询的权限，可以获得模型的 softmax 输出或者最后的分类标签。本次实验暂未实现对模型的黑盒攻击。

3 实验方法

3.1 实验架构

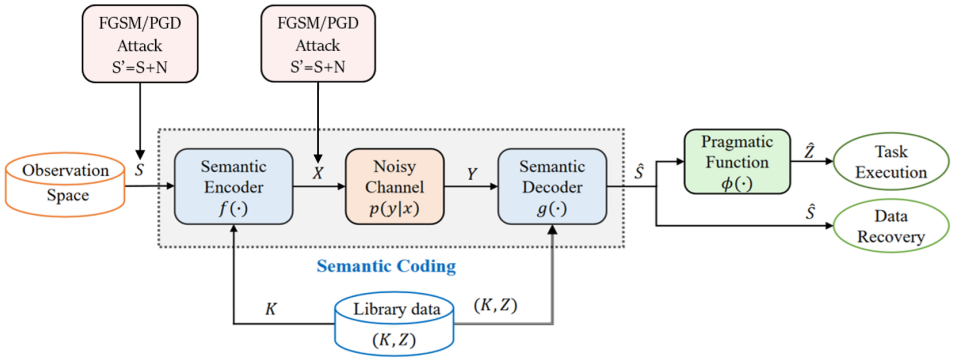


Figure 2: frame of our experiment

3.2 实验流程

实验总体分两阶段进行。第一阶段是实现一个语义通信系统，由编码器、噪声信道、译码器以及经验数据库组成。第二阶段是基于白盒设定实现 FGSM 和 PGD 对语义通信系统的对抗样本攻击。理论上，可以在两个阶段进行对抗样本攻击，由于方法一致和时间原因，且为了方便查看添加扰动前后的效果，仅拿进入编码器前的对抗样本攻击进行实验。

3.3 实验设定

Dataset 使用 MNIST 数据集 (LeCun et al. (1998))，训练集由来自 250 个不同人手写的数字构成，测试集也是同样比例的手写数字数据，但保证了测试集和训练集的作者集不相交。MNIST 数据集一共有 7 万张图片，其中 6 万张是训练集，1 万张是测试集。每张图片是 28×28 的 $0-9$ 的灰度手写数字图片组成，每个图片是黑底白字的形式。

Pragmatic Function 接收端已知的语义提取函数使用 MLP 结构进行训练，由四层全连接层构成，目标是准确分类手写数字。epoch 为 20，训练集 batchsize 为 64，如图 Figure 3，最终训练集准确率为 88.95%，测试集准确率为 89.52%。

- fc1: $(28 \times 28, 500)$
- fc2: $(500, 250)$
- fc3: $(250, 125)$
- fc4: $(125, 10)$

```
Downloading http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz
Downloading http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz to ./dataset/mnist/MNIST/raw/t10k-labels-idx
1-ubyte.gz
5120it [00:00, 21957910.51it/s]
Extracting ./dataset/mnist/MNIST/raw/t10k-labels-idx1-ubyte.gz to ./dataset/mnist/MNIST/raw

Processing...
/data/home/JingJing_Wang/anaconda3/envs/purifier/lib/python3.8/site-packages/torchvision/datasets/mnist.py:502: Use
rWarning: The given NumPy array is not writable, and PyTorch does not support non-writable tensors. This means writ
ing to this tensor will result in undefined behavior. You may want to copy the array to protect its data or make it
 writable before converting it to a tensor. This type of warning will be suppressed for the rest of this program. (
Triggered internally at ../torch/csrc/utils/tensor_numpy.cpp:199.)
  return torch.from_numpy(parsed.astype(m[2], copy=False)).view(*s)
Done!
epoch: 0, Train Loss: 2.282707, Train Acc: 0.205707, Eval Loss: 2.256821, Eval Acc: 0.353738
epoch: 1, Train Loss: 2.225860, Train Acc: 0.373351, Eval Loss: 2.180201, Eval Acc: 0.429490
epoch: 2, Train Loss: 2.111731, Train Acc: 0.440382, Eval Loss: 2.007900, Eval Acc: 0.484177
epoch: 3, Train Loss: 1.850089, Train Acc: 0.520622, Eval Loss: 1.634914, Eval Acc: 0.574169
epoch: 4, Train Loss: 1.427867, Train Acc: 0.618770, Eval Loss: 1.208348, Eval Acc: 0.670688
epoch: 5, Train Loss: 1.068630, Train Acc: 0.709288, Eval Loss: 0.919588, Eval Acc: 0.737836
epoch: 6, Train Loss: 0.839458, Train Acc: 0.766624, Eval Loss: 0.740670, Eval Acc: 0.784415
epoch: 7, Train Loss: 0.700795, Train Acc: 0.798457, Eval Loss: 0.632199, Eval Acc: 0.811511
epoch: 8, Train Loss: 0.612289, Train Acc: 0.822578, Eval Loss: 0.562266, Eval Acc: 0.834355
epoch: 9, Train Loss: 0.556886, Train Acc: 0.836237, Eval Loss: 0.519724, Eval Acc: 0.843157
epoch: 10, Train Loss: 0.516501, Train Acc: 0.847215, Eval Loss: 0.484455, Eval Acc: 0.854727
epoch: 11, Train Loss: 0.486270, Train Acc: 0.856726, Eval Loss: 0.456989, Eval Acc: 0.867188
epoch: 12, Train Loss: 0.463527, Train Acc: 0.862790, Eval Loss: 0.438265, Eval Acc: 0.870847
epoch: 13, Train Loss: 0.445285, Train Acc: 0.869470, Eval Loss: 0.419008, Eval Acc: 0.879055
epoch: 14, Train Loss: 0.429450, Train Acc: 0.873534, Eval Loss: 0.405275, Eval Acc: 0.880340
epoch: 15, Train Loss: 0.414647, Train Acc: 0.877482, Eval Loss: 0.394804, Eval Acc: 0.884098
epoch: 16, Train Loss: 0.401677, Train Acc: 0.881930, Eval Loss: 0.387342, Eval Acc: 0.885384
epoch: 17, Train Loss: 0.394029, Train Acc: 0.884162, Eval Loss: 0.375387, Eval Acc: 0.890922
epoch: 18, Train Loss: 0.383545, Train Acc: 0.886794, Eval Loss: 0.364880, Eval Acc: 0.892603
epoch: 19, Train Loss: 0.376688, Train Acc: 0.889592, Eval Loss: 0.359979, Eval Acc: 0.895273
```

Figure 3: MLP MNIST model for pragmatic function

Semantic Coding Network 语义编码网络由编码器、噪声信道和解码器组成。编码器和解码器使用 MLP 结构，均为一层全连接层，根据数据压缩率进行设定。信道假设为 AWGN 信道，输出为 $Y = X + N$，X 为输入数据，N 为信道噪声，Y 为接收端接收到的数据。compression rate 取 0.5，epoch 为 500，训练集 batchsize 为 64，如图 Figure 4，训练准确率为 90.39%，测试集准确率为 90.42%。PSNR 为 17.77。

```
epoch: 492, Train Loss: 0.208382, Train Acc: 0.903735, Eval Loss: 0.203191, Eval Acc: 0.903481, PSNR: 19.386379
epoch: 493, Train Loss: 0.208001, Train Acc: 0.904218, Eval Loss: 0.204070, Eval Acc: 0.902591, PSNR: 19.607747
epoch: 494, Train Loss: 0.208112, Train Acc: 0.903785, Eval Loss: 0.203250, Eval Acc: 0.903184, PSNR: 19.551376
epoch: 495, Train Loss: 0.208141, Train Acc: 0.903501, Eval Loss: 0.203651, Eval Acc: 0.903877, PSNR: 18.458026
epoch: 496, Train Loss: 0.207590, Train Acc: 0.904301, Eval Loss: 0.202621, Eval Acc: 0.903877, PSNR: 19.234422
epoch: 497, Train Loss: 0.208155, Train Acc: 0.903551, Eval Loss: 0.203450, Eval Acc: 0.902987, PSNR: 21.034420
epoch: 498, Train Loss: 0.207671, Train Acc: 0.904634, Eval Loss: 0.203769, Eval Acc: 0.904470, PSNR: 19.966714
epoch: 499, Train Loss: 0.207801, Train Acc: 0.903985, Eval Loss: 0.203219, Eval Acc: 0.904272, PSNR: 17.773138
(purifier) JingJing_Wang@aisec-01:~/Semantic-Communication-Systems/semantic_extraction$
```

Figure 4: Semantic Coding Network under 0.5 compression rate

FGSM Attack 对于 FGSM 攻击，固定最大攻击次数 (iterations) 为 10，在压缩比为 0.5 的情况下，分别对添加扰动的大小 $\epsilon = 0, 0.1, 0.2, 0.3$ 进行四组试验。为了方便和 PGD 对比，另外设置了攻击次数为 1 的一组实验。

PGD Attack 对于 PGD 攻击，取定最大攻击次数为 $\frac{\epsilon}{\alpha}$，其中 α 表示每次添加的扰动的大小，一般远小于 FGSM 的扰动大小，ϵ 表示添加的扰动的阈值，在压缩比为 0.5 的情况下，$\epsilon = 0.1, 0.2, 0.3, 0.4$，$\alpha = 0.01, 0.02$ 情形下进行八组实验。

4 关键代码实现

4.1 semantic coding network

```python
class MLP(nn.Module):
    # coders
    def __init__(self):
        super(MLP, self).__init__()
        self.fc1 = nn.Linear(28 * 28, channel)
        self.fc2 = nn.Linear(channel, 28 * 28)
    def forward(self, x):
        # encoder
        x = self.fc1(x)
        # ......
        # some other operations

        # add noise
        x_np = x.detach().numpy()
        out_square = np.square(x_np)
        aver = np.sum(out_square) / np.size(out_square)
        snr = 10  # dB
        aver_noise = aver / 10 ** (snr / 10)
        noise = np.random.random(size=x_np.shape) * np.sqrt(aver_noise)
        # Y=X+N
        x_np = x_np + noise
        x = torch.from_numpy(x_np)
        x = x.to(torch.float32)

        # decoder
        x = self.fc2(x)
        return x
```

4.2 FGSM Attack

```python
data_grad = img.grad.data      #计算输入的导数
sign_data_grad = data_grad.sign()  #符号函数
img = img.detach()
img += epsilon * sign_data_grad
img = torch.clamp(img, 0, 1)
```

4.3 PGD Attack

```python
sign_data_grad = data_grad.sign()  #符号函数
img = img.detach()
adv_img = img + alpha * sign_data_grad
eta = torch.clamp(adv_img-img, min=-epsilon, max=epsilon)
```

```
5 img = torch.clamp(img + eta, min=0, max=1).detach()
```

5 实验结果

5.1 FGSM Attack

在 FGSM 攻击中，如果攻击能在 10 次内完成，就可以认为攻击是成功的，反之则认为攻击失败。至于攻击上限为什么定为 10 次，这是基于另外一组实验，当攻击次数上限继续增加时，攻击成功率并不会显著增加，在这里就不再详细展开。为了节省时间，选定 10 作为攻击次数的上限。注意区别于 PGD，FGSM 的攻击尽管有多轮迭代，但每一轮都以一个较大的扰动 ϵ 发生改变。

与此同时，通过改变 fgsm 的 epsilon 大小，可以发现 epsilon 的大小与对抗样本图像的干扰都和接收端恢复的图像的的清晰度成正比，epsilon 越大，图像的干扰越明显，攻击的效果一般也相对更好。

Attack Accuracy and Total Attack Times

首先对不同 ϵ 下的攻击成功率和总攻击次数进行了探究（原始实验图片放至附录中）。得到的攻击成功率和总攻击次数如下表。

Table 1: attack results under FGSM, with iteration=10, $\epsilon = 0, 0.1, 0.2, 0.3$

	$\epsilon = 0$	$\epsilon = 0.1$	$\epsilon = 0.2$	$\epsilon = 0.3$
attack accuracy	0.3827	0.8882	**0.9251**	**0.9292**
total attack times	68649	44125	36300	33462
average attack times	6.86	4.41	3.63	3.34

从表格中可以发现，随着 ϵ 的逐渐增大，攻击准确率逐渐提高，但当 ϵ 接近 0.2 时，攻击准确率提高的速度放缓，趋于收敛，说明再加大扰动，只能影响小部分图片的攻击成功率。此外，总攻击次数和平均攻击次数随着 ϵ 的增大而减小，也即随着扰动大小的增加，增加扰动的次数较少，对于图片的少数较大扰动足够让目标接收端分类错误。

值得注意的是，当 $\epsilon = 0$ 时，理论上没有对原图添加任何扰动，但获得的攻击准确率却不为 0%。我们将这点解释成，由于接收端的语义分类模型以及语义编码解码网络本身的准确率不是很高，基本处在 90% 上下，导致存在部分图片本身就被分类错误。另一方面，则是由于在处理过程中，对图片进行了一些 clamp 或者是其他的裁剪操作所引入的累计误差。具体原因留待以后分析。

结合上面两图我们可以得出结论，对于我们的模型，对抗样本的干扰越大，能够更快地、更准确地完成攻击。但是单纯通过提高干扰程度带来的提高是有限的，当干扰大到一定程度的时候，对攻击速度和准确率的提高就不太多了。

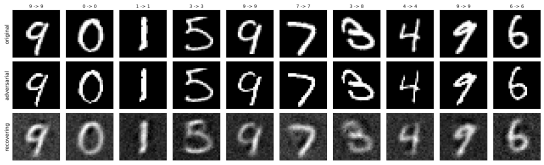
为了方便和 PGD 进行对比，另外进行了一组 $iteration = 1$(攻击次数恒定为 1) 的 FGSM 实验，结果如下表。

Table 2: attack results under FGSM, with iteration=1, $\epsilon = 0, 0.1, 0.2, 0.3, 0.4$

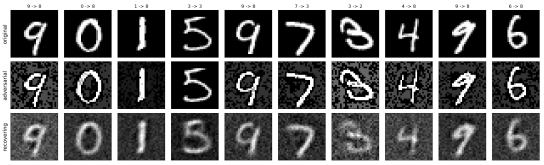
	0	0.1	0.2	0.3	0.4
Attack Accuracy	0.1286	0.1277	0.1289	0.1290	**0.1295**

Adversarial Samples and Recovered Images

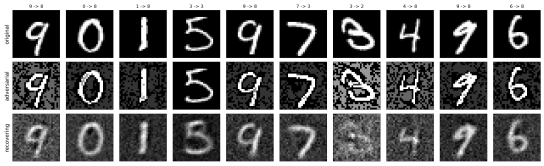
Figure 5显示了在 $\epsilon = 0, 0.1, 0.2, 0.3$ 的情形下进行 FGSM 攻击的图片效果。第一行是 MNIST 的原图，第二行是经过 FGSM 攻击处理后的图片，第三行是经过语义通信系统接收端解码出来的图片。



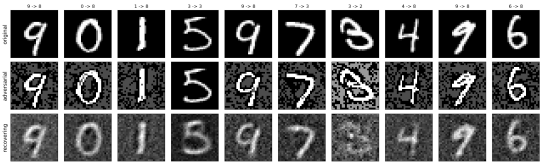
(a) FGSM: $\epsilon = 0$



(b) FGSM: $\epsilon = 0.1$



(c) FGSM: $\epsilon = 0.2$



(d) FGSM: $\epsilon = 0.3$

Figure 5: First row the original picture, second row the picture after FGSM attack, third row the picture recovered by the receiver, under $\epsilon = 0, 0.1, 0.2, 0.3$ with MNIST dataset

其中，Figure 5(a)给出的是 ϵ=0 的情况，此时可以认为生成的样本几乎没有添加干扰，从结果来看，绝大部分样本都被识别成功。当 ϵ 增加到 0.1 时，对抗样本增加了一定的干扰。此时接受端还原出来的图片并没有产生特别大的区别，但是原先能够被正确识别的许多数字现在却产生了错误。例如原先数字 9 能够被正确识别，现在却被识别成了 8；原先能够被正确识别的的 7 也被识别成了 3。

继续增加 ϵ 的大小，我们可以看出，生成的对抗样本明显增加了更多的干扰，此时接受端产生的图片相比原图的差异更大了，识别也是同样出错了。

观察识别的结果，我们发现各种数字识别出错的方向是一致的，就是说，原先 9 被错误识别成了 8，在增大干扰之后仍然是被误识别为 8。而且有意思的是，在攻击之后，有许多数字都被误认为了 8，这应该是本次实验实现的语义通信网络的特性。

Deep Exploration with Class-based Granularity

为了进一步探究 FGSM 对于样本的转换趋向，我们把粒度缩小到类，从类的视角来研究不同类的图片经过 FGSM 处理后的目标转换趋向。如下图所示。

横坐标为经过 FGSM 转换后接收端的分类结果，每一种颜色代表某一类的真实标签。

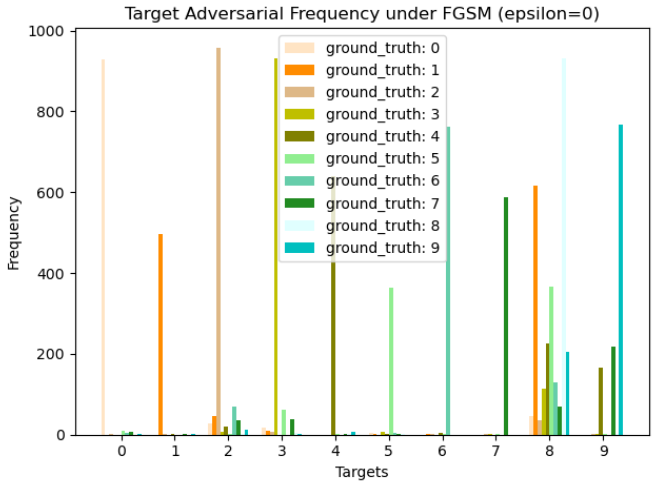


Figure 6: class granularity transform target under FGSM($\epsilon = 0$)

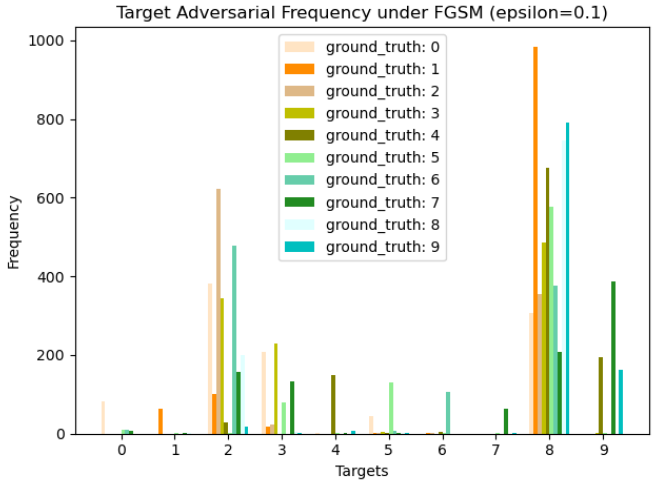


Figure 7: class granularity transform target under FGSM($\epsilon = 0.1$)

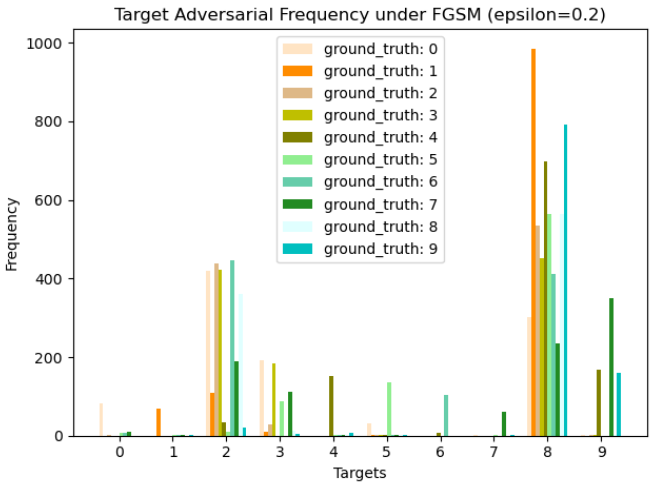


Figure 8: class granularity transform target under FGSM($\epsilon = 0.2$)

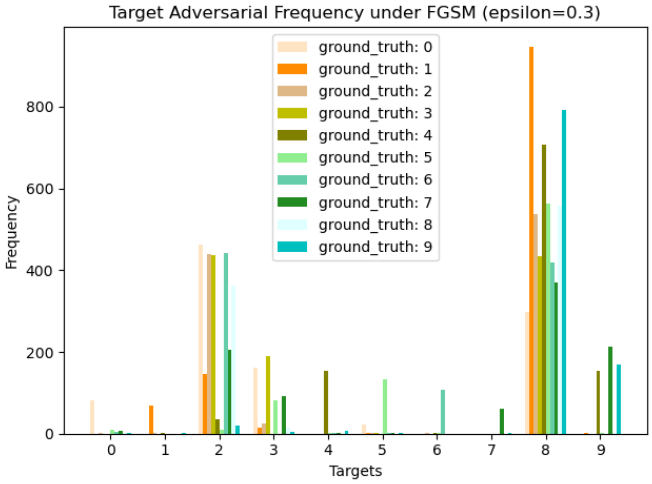


Figure 9: class granularity transform target under FGSM($\epsilon = 0.3$)

从 Figure 6可以看出，当不对图片添加任何额外噪声时，由于模型本身准确率的原因，对于第 1、4、5 类识别准确率较低，容易把数字 1、5 分类成数字 8。由此带来了不用攻击本身就存在的基准成功率。

进而比较 Figure 7和 Figure 8可以发现，随着 ϵ 的增加，第 2 类和第 8 类正确分类的图片数量显著下降。其它类别正确分类的数目也都有下降。

综合四张图片，可以看出，同一个类别在 FGSM 之后，倾向于转换成另外的某个固定的类。譬如，大部分类别都容易转换成第 8 类，而第 6 类容易被转换成第 2 类。猜测是由于类与类之间决策边界的远近决定的。后续探索可以使用 t-SNE 等方法对样本空间进行降维。

5.2 PGD Attack

Attack Accuracy and Total Attack Times

在 PGD 攻击中，我们改变了 α 与 ϵ 的数值，并对攻击的成功率和平均攻击次数进行了分析。其中 α 代表对抗样本每次移动的步长，而 ϵ 代表对抗样本对原图片干扰的最大程度。也就是说，α 越小代表每次梯度上升的幅度越小，而 ϵ 则决定了对抗样本干扰程度的上限。由于干扰程度不能超过 ϵ, 所以攻击的迭代次数不需要超过 $\frac{\epsilon}{\alpha}$。

Table 3: attack results under PGD, with $\epsilon = 0.1, 0.2, 0.3, 0.4, \alpha = 0.01, 0.02$, Average Times lines in the form of actual average times/attack iteration threshold($\frac{\epsilon}{\alpha}$)

		$\epsilon = 0.1$	$\epsilon = 0.2$	$\epsilon = 0.3$	$\epsilon = 0.4$
$\alpha = 0.01$	Accuracy	0.4509	0.5795	0.6966	**0.7665**
	Total Times	66527	115792	152202	179108
	Average Times	6.65/10	11.57/20	15.22/30	17.91/40
$\alpha = 0.02$	Accuracy	0.4266	0.5533	0.6814	0.7582
	Total Times	37210	63410	83045	97380
	Average Times	3.72/5	6.34/10	8.30/15	9.73/20

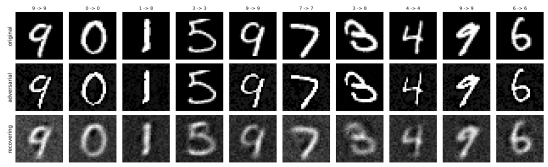
从表格中我们可以发现，在同一 ϵ 下，α 的值越小，攻击的准确率越高，这说明步长更小时，对抗样本更容易朝着使得原图片出错的方向移动；这也体现在了平均循环次数上，α 更小时，对抗样本可以更有效率地找到合适的干扰。不过步长更小使得我们需要更多的“步数”，也就是我们需要更多的循环次数才能得到有效的对抗样本。

在 α 保持一致的情况下，ϵ 的数值越大，攻击的准确率越高，这也符合我们的直觉，实验数据说明在同样的步长的情况下，干扰程度上限的提高能够提高攻击的准确率。上限的提高也带来了循环次数的增多，部分对抗样本需要“走更远的路”才能到达有效的干扰效果。

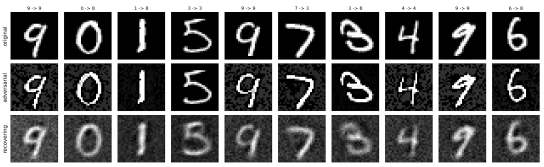
我们可以和 FGSM 攻击进行对比，在现实的例子中，我们往往不能让对抗样本相比于原图的干扰过大，反映在 FGSM 中就是只能增加一次干扰，如 Table 2所示，我们可以发现在确定干扰上限的情况下，PGD 有更好的表现。

Adversarial Samples and Recovered Images

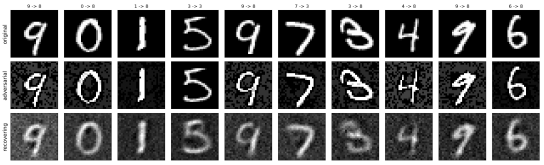
在 PGD 攻击中，我们也观察了不同 α 和 ϵ 的原始图片、对抗样本和接收端恢复的图片，如下面八张图片所示。容易发现，相对于 FGSM 攻击，PGD 对于原图的扰动相对较小，不容易看出和原图之间的差别，更能符合现实攻击的需要，即在扰动较小的情况下达到一个可观的攻击准确率。而如果使用 FGSM 攻击，难以寻找到较小扰动下的最优解。PGD 虽然引入了 α 的较小步长，导致攻击时间大大增加，却可以达到更好的攻击效果。



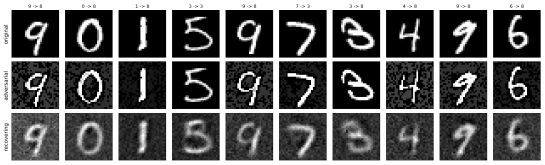
(a) PGD: $\alpha = 0.01$ $\epsilon = 0.1$



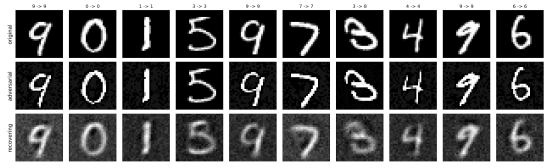
(b) PGD: $\alpha = 0.01$ $\epsilon = 0.2$



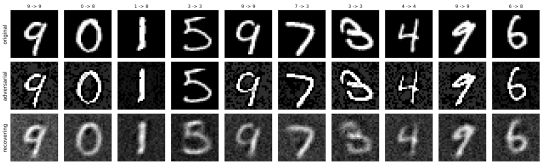
(c) PGD: $\alpha = 0.01$ $\epsilon = 0.3$



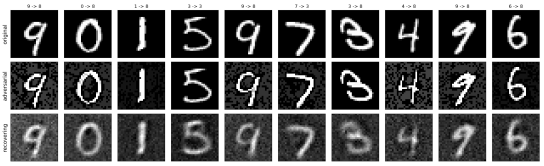
(d) PGD: $\alpha = 0.01$ $\epsilon = 0.4$



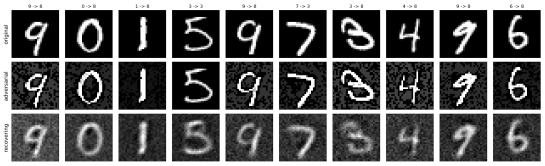
(e) PGD: $\alpha = 0.02$ $\epsilon = 0.1$



(f) PGD: $\alpha = 0.02$ $\epsilon = 0.2$



(g) PGD: $\alpha = 0.02$ $\epsilon = 0.3$



(h) PGD: $\alpha = 0.02$ $\epsilon = 0.4$

6 思考与心得

本次实验围绕 FGSM 和 PGD 对搭建的语义通信系统展开对抗样本攻击，由于是白盒情形，攻击者可以获取较多的模型信息，攻击成功率较高，相对比较容易。在 FGSM 和 PGD 的对比中，我们发现 PGD 需要以较多的攻击轮数，也即较高的时间来换取较小扰动小的高准确率。由于实验设备和时间受限，我们没有再减小 PGD 的 α 来进行探索。相比于进行 10 轮攻击的 FGSM，PGD 的效果有待提高，却显著高于仅用 1 轮攻击的 FGSM。我们将其解释为，PGD 的探索轮数不够，以及所搭建的语义通信模型包括其后的语义分类函数只用了简单的 MLP 来实现，梯度特征较为简单。

此外，对于对抗样本的转换特征。我们观察到有明显的类别粒度倾向，其中的原因可以通过降维来更清晰的揭示，也留待日后探索。

对于语义通信系统的黑盒攻击，有一个初步的设想但尚未实现，其基本思路就是基于 shadow model（影子模型）或者称为 reference model。假定攻击者知道所有的训练方式和模型结构，但不知道具体的训练数据集以及模型参数，攻击可以从原始数据集中抽

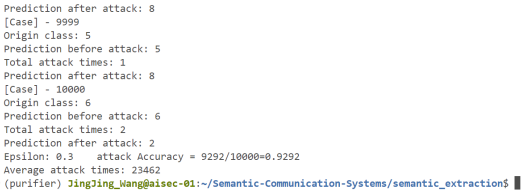
取若干张图片训练一个影子语义通信系统。然后以该影子模型为攻击目标进行对抗样本的制作，再去攻击原始模型。理论上，由于 MLP 结构的单一性以及 MNIST 数据集的简洁性，此种方式也能达到客观的攻击成功率，具体成效留待以后探索。

7 Appendix

(i) FGSM: $\epsilon = 0$

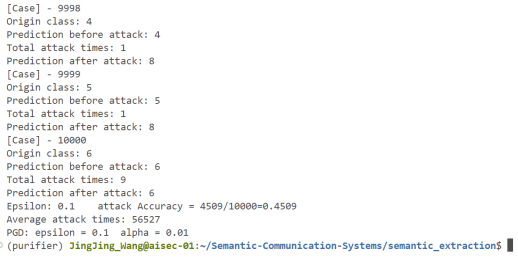
(j) FGSM: $\epsilon = 0.1$

(k) FGSM: $\epsilon = 0.2$



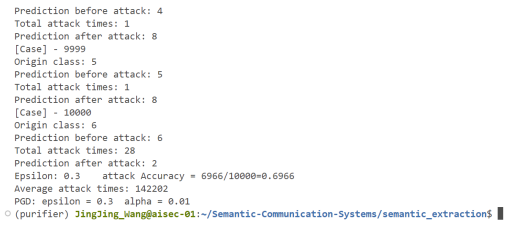
(l) FGSM: $\epsilon = 0.3$

Figure 10: FGSM attack accuracy and total attack times under different epsilon

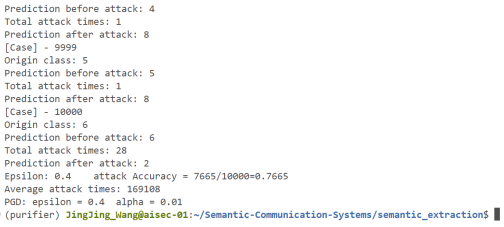


(a) PGD: $\alpha = 0.01$ $\epsilon = 0.1$

(b) PGD: $\alpha = 0.01$ $\epsilon = 0.2$

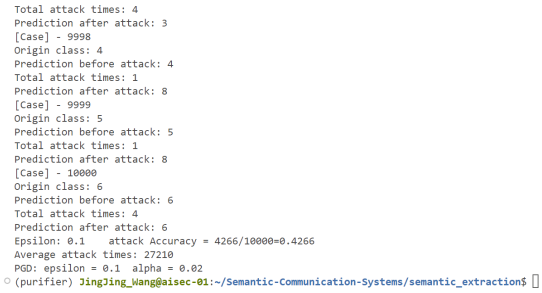


(c) PGD: $\alpha = 0.01$ $\epsilon = 0.3$

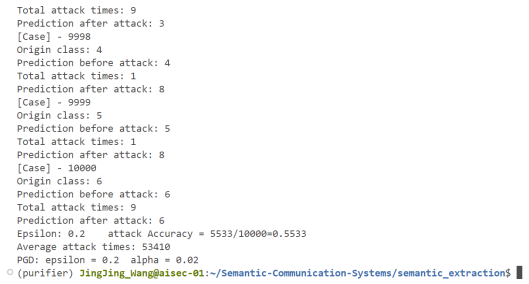


(d) PGD: $\alpha = 0.01$ $\epsilon = 0.4$

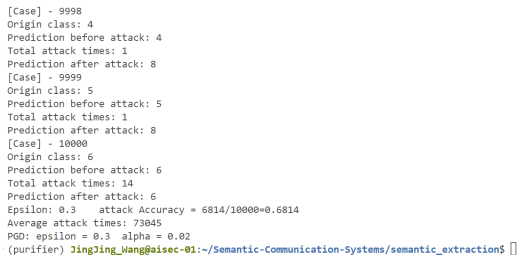
Figure 11: PGD attack accuracy and total attack times under different epsilon of $\alpha = 0.01$



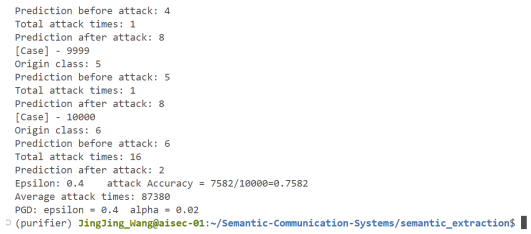
(a) PGD: $\alpha = 0.02$ $\epsilon = 0.1$



(b) PGD: $\alpha = 0.02$ $\epsilon = 0.2$



(c) PGD: $\alpha = 0.02$ $\epsilon = 0.3$



(d) PGD: $\alpha = 0.02$ $\epsilon = 0.4$

Figure 12: PGD attack accuracy and total attack times under different epsilon of $\alpha = 0.02$

References

Goodfellow, I. J., J. Shlens, and C. Szegedy (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*(11), 2278–2324.

Madry, A., A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Zhang, H., S. Shao, M. Tao, X. Bi, and K. B. Letaief (2022). Deep learning-enabled semantic communication systems with task-unaware transmitter and dynamic data. *arXiv preprint arXiv:2205.00271*.