Lecture 12 Noisy SGD and Deep Learning with DP

Yu-Xiang Wang

COMPUTER SCIENCE
UC SANTA BARBARA
Computing. ReInvented.

Administrative notes

- Project proposal feedback in Gradescope!
  - Midterm report due next week
- HW2 Q4 almost ready… stay tuned for the Piazza announcement
  - This lecture might be useful for you to see how learning rates are chosen

Recap: Last lecture

- Objective perturbation vs Output perturbation
- Gradient Descent and Stochastic Gradient Descent
  - Convergence analysis
- NoisyGD mechanism

Recap: Compare the excess **empirical** risk of Output/Objective Perturbation

	Lipschitz losses	Smooth losses	Smooth / Lipschitz GLM
Output Pert	$\frac{d^{1/4}L\|\theta^*\|\log(\frac{1}{\delta})^{1/4}}{n^{1/2}\epsilon^{1/2}}$	$\frac{d^{1/3}\beta^{1/3}L^{2/3}\|\theta^*\|^{4/3}\log(\frac{1}{\delta})^{1/3}}{n^{2/3}\epsilon^{2/3}}$	Same as left
ObjPert	Not applicable	$\frac{dL\|\theta^*\|\sqrt{\log(\frac{1}{\delta})}}{n\epsilon}$	$\frac{\sqrt{d}L\|\theta^*\|\sqrt{\log(\frac{1}{\delta})}}{n\epsilon}$

Lower order terms and dependence on β hidden.

Recap: What are not quite satisfactory?

- Require the loss to be twice differentiable
  - Convex losses need not be even differentiable
- We did not handle the constrained convex ERM
- They do not handle non-convex ERM problems, e.g., those that arise when optimizing deep neural networks

Recap: Renyi Differential Privacy and Concentrated Differential Privacy

- We say that a mechanism satisfies (α, ϵ)-Renyi DP, if

$$D_\alpha(\mathcal{M}(x)\|\mathcal{M}(x')) \leq \epsilon$$

- We say a mechanism satisfies ρ-zCDP, if

$$D_\alpha(\mathcal{M}(x)\|\mathcal{M}(x')) \leq \rho\alpha, \forall\alpha > 1$$

- Gaussian mechanism satisfies:

- Converting to approximate DP:

$$(\epsilon, \alpha)\text{-RDP implies } (\epsilon(\alpha) + \tfrac{\log(1/\delta)}{\alpha-1}, \delta)\text{-DP}$$

$$\textit{If } M \textit{ provides } \rho\textit{-zCDP, then } M \textit{ is } (\rho + 2\sqrt{\rho\log(1/\delta)}, \delta)\textit{-DP.}$$

Recap: Noisy Gradient Descent Mechanism

- The algorithm:

- Privacy analysis:
  - A composition of T Gaussian mechanisms

Today

- Utility analysis of NoisyGD
  - Applying convergence of SGD from Ghadimi and Lan to analyze NoisyGD mechanism
- NoisySGD and privacy amplification by sampling
- Application to Deep Learning with Differential Privacy

Readings

- Convergence of SGD
  - Smooth/nonconvex and convex case: Ghadimi and Lan: https://arxiv.org/pdf/1309.5549.pdf
  - Strongly convex case: https://arxiv.org/pdf/1212.2002v2.pdf
  - Convex / nonsmooth case: My handwritten notes from 292F: https://sites.cs.ucsb.edu/~yuxiangw/classes/CS292F-2020Spring/Lectures/notes_lecture8.pdf

- NoisySGD: Bassily et al. https://arxiv.org/abs/1405.7085

- Deep Learning with DP: Abadi et al. https://arxiv.org/abs/1607.00133

- Amplification by sampling in RDP: https://arxiv.org/abs/1808.00087

Convergence guarantee of NoisyGD for nonconvex / smooth problems

- Convergence bound from last time:

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\left[\|\nabla f(x_t)\|^2\right] \leq \frac{2(f(x_1)-f^*)}{T\eta} + \eta n\beta d\sigma^2$$

  - Choice of learning rate: $\min\{\frac{1}{n\beta}, \frac{\sqrt{2(f(x_1)-f^*)}}{\sqrt{n\beta d\sigma^2 T}}\}$
- Final utility bound:

Convergence guarantee of NoisyGD for convex /smooth problems

- Similar analysis, not covered (Read Ghadimi and Lan)

$$\mathbb{E}\left[\frac{1}{T}\sum_t (f(x_t) - f^*)\right] \leq \frac{\|x_1 - x^*\|^2}{T\eta} + \eta d\sigma^2$$

  - Choice of learning rate $\min\{\frac{1}{n\beta}, \frac{\|x_1 - x^*\|}{\sqrt{d\sigma^2 T}}\}$
- Final utility bound:

Convergence guarantee of NoisyGD for convex /Lipschitz problems

- Similar analysis, not covered (Read my notes from CS292F Convex Optimization Lecture 8)

$$\mathbb{E}\left[\frac{1}{T}\sum_{t}(f(x_t)-f^*)\right] \leq \frac{\|x_1-x^*\|^2}{T\eta} + \eta(\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T}\|\partial f(x_t)\|^2\right] + d\sigma^2)$$

  - Learning rate chosen optimally
- Final utility bound

Convergence guarantee of NoisyGD for strongly convex / Lipschitz problems

- Convergence is even faster!
  - Learning rate: $\eta_t = \frac{1}{\lambda t}$

$$\mathbb{E}\left[f(\frac{1}{T}\sum_{t=1}^{T} x_t)\right] - f(x^*) \leq \frac{n^2L^2 + d\sigma^2}{2\lambda T}(1 + \log T)$$

  - Learning rate: $\eta_t = \frac{2}{\lambda(t+1)}$

$$\mathbb{E}\left[f(\frac{2}{T(T+1)}\sum_{t=1}^{T} t x_t)\right] - f(x^*) \leq \frac{4(n^2L^2 + d\sigma^2)}{\lambda(T+1)}$$

Lacoste-Julien, Schmidt, Bach: https://arxiv.org/pdf/1212.2002v2.pdf

NoisyGD and strongly convex problems

$$\mathbb{E}\left[f(\frac{2}{T(T+1)}\sum_{t=1}^{T} t x_t)\right] - f(x^*) \leq \frac{4(n^2L^2 + d\sigma^2)}{\lambda(T+1)}$$

- Utility analysis under zCDP:

Checkpoint: NoisyGD summary

	Lipschitz + convex	Lipschitz + Smooth + convex	Smooth + Lipschitz + convex + GLM
Output Pert	$\frac{d^{1/4}L\|\theta^*\|\log(\frac{1}{\delta})^{1/4}}{n^{1/2}\epsilon^{1/2}}$	$\frac{d^{1/3}\beta^{1/3}L^{2/3}\|\theta^*\|^{4/3}\log(\frac{1}{\delta})^{1/3}}{n^{2/3}\epsilon^{2/3}}$	Same as left
ObjPert	Not applicable	$\frac{dL\|\theta^*\|\sqrt{\log(\frac{1}{\delta})}}{n\epsilon}$	$\frac{\sqrt{d}L\|\theta^*\|\sqrt{\log(\frac{1}{\delta})}}{n\epsilon}$
NoisyGD	$\frac{\sqrt{d}L\|\theta^*\|\sqrt{\log(\frac{1}{\delta})}}{n\epsilon}$	$\frac{\sqrt{d}L\|\theta^*\|\sqrt{\log(\frac{1}{\delta})}}{n\epsilon}$	$\frac{\sqrt{d}L\|\theta^*\|\sqrt{\log(\frac{1}{\delta})}}{n\epsilon}$

Lower order terms and dependence on β hidden.

	Lipschitz + Strongly convex	Lipschitz + Smooth + Nonconvex
NoisyGD	$\frac{dL^2\log(1/\delta)}{n\lambda\epsilon^2}$	$\frac{\sqrt{n\beta dL^2(f(\theta_1)-f^*)\log(1/\delta)}}{n\epsilon}$

Stationary point convergence

The advantage of NoisyGD

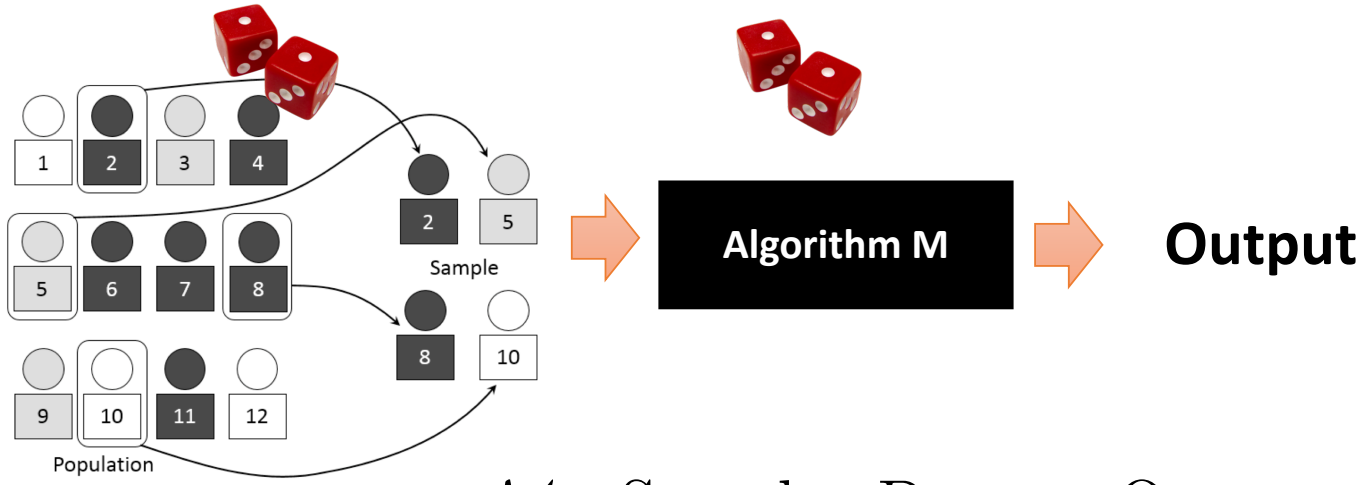
- It is more generally applicable
- Results in stronger guarantees
- Do not require exact optimal solution

Computational Complexity of NoisyGD

- General pattern: First term goes to 0, second term dominates with large T. (How large does T need to be?)
  - Convex + Lipschitz
  - Convex + Smooth
  - Strongly convex + Lipschitz

Each iteration requires n incremental gradient evaluation!

Privacy Amplification by Sampling



$$\mathcal{M} \circ \text{Sample} : \text{Data} \rightarrow \text{Output}$$

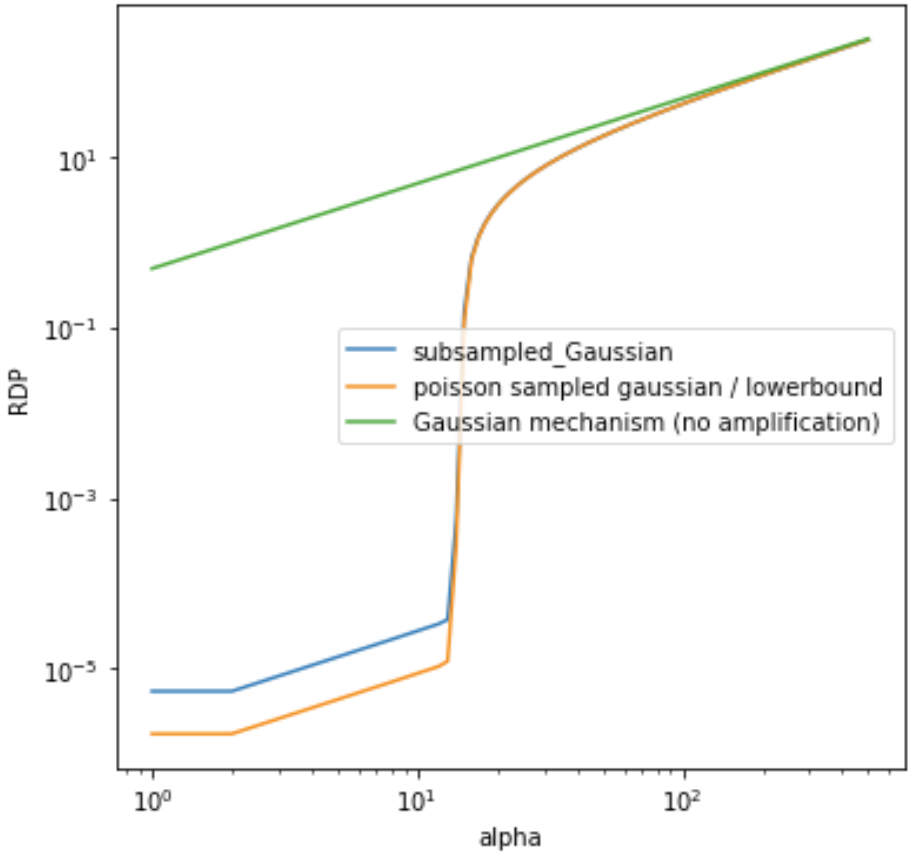
Subsampling Lemma: If M obeys (Ɛ,δ)-DP, then M ∘ Subsample obeys that (Ɛ',δ')-DP with $\delta' = \gamma\delta$

$$\epsilon' = \log(1 + \gamma(e^{\epsilon} - 1)) = O(\gamma\epsilon)$$

Two different sampling schemes for privacy amplification

- Poisson Sampling vs Sampling without Replacement

The Renyi DP of sampled mechanisms



The Noisy Stochastic Gradient Descent Mechanism (NoisySGD)

- Privacy analysis:
  - A composition of T **subsampled** gaussian mechanism.

- RDP simply adds up

The Noisy Stochastic Gradient Descent Mechanism (NoisySGD)

- Utility analysis:
  - What is the gradient estimates?
  - Same bounds as before, but noise gets larger
  - Computational consideration

Remainder of the lecture

- A survey of differentially private deep learning

- Reference:
  - NoisySGD: Abadi et al. (2016)
  - PATE: Papernot et al. (2018), PrivateKNN: Zhu et al. (2020)

  - Autodp tutorial on private deep learning: https://github.com/yuxiangw/autodp/blob/master/tutorials/tutorial_private_deep_learning.ipynb

Deep Learning with Differential Privacy

- Main challenge: It is difficult to enforce global sensitivity

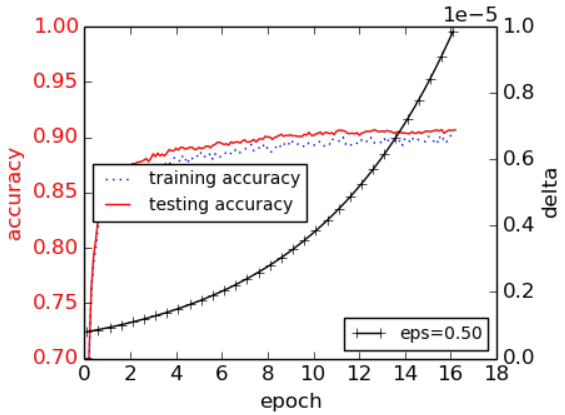
- Abadi et al: Per-example Gradient Clipping

- Other tricks:
  - Use small neural networks
  - Private-PCA to reduce dimension in the original data

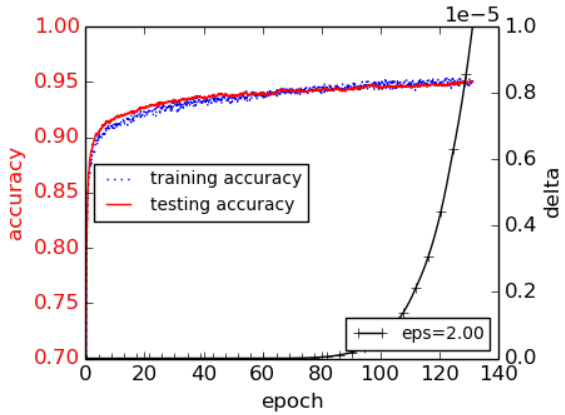
What does Deep Learning with Differential Privacy look like in autodp?

```
from autodp.autodp_core import Mechanism
from autodp.transformer_zoo import Composition
from autodp import mechanism_zoo, transformer_zoo

class NoisySGD_mech(Mechanism):
    def __init__(self,prob,sigma,niter,name='NoisySGD'):
        Mechanism.__init__(self)
        self.name=name
        self.params={'prob':prob,'sigma':sigma,'niter':niter}

        # create such a mechanism as in previously
        subsample = transformer_zoo.AmplificationBySampling() # by default this is using poisson sampling
        mech = mechanism_zoo.GaussianMechanism(sigma=sigma)
        prob = prob
        # Create subsampled Gaussian mechanism
        SubsampledGaussian_mech = subsample(mech,prob,improved_bound_flag=True)

        # Now run this for niter iterations
        compose = transformer_zoo.Composition()
        mech = compose([SubsampledGaussian_mech],[niter])

        # Now we get it and let's extract the RDP function and assign it to the current mech being constructed
        rdp_total = mech.RenyiDP
        self.propagate_updates(rdp_total,type_of_update='RDP')
```

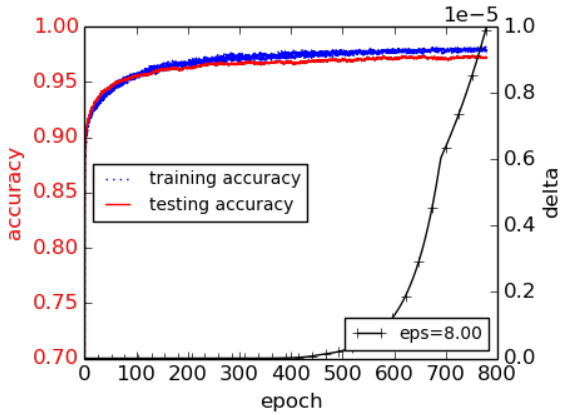
Empirical results on MNIST



(1) Large noise



(2) Medium noise

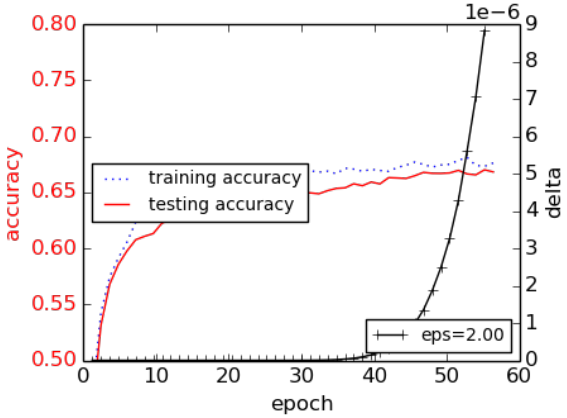


(3) Small noise

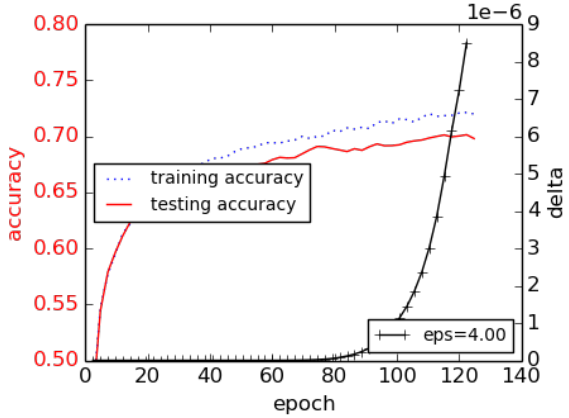
Figure 3: Results on the accuracy for different noise levels on the MNIST dataset. In all the experiments, the network uses 60 dimension PCA projection, 1,000 hidden units, and is trained using lot size 600 and clipping threshold 4. The noise levels (σ, σ_p) for training the neural network and for PCA projection are set at (8, 16), (4, 7), and (2, 4), respectively, for the three experiments.

Abadi et al. (2016) Deep Learning with Differential Privacy: https://arxiv.org/pdf/1607.00133.pdf

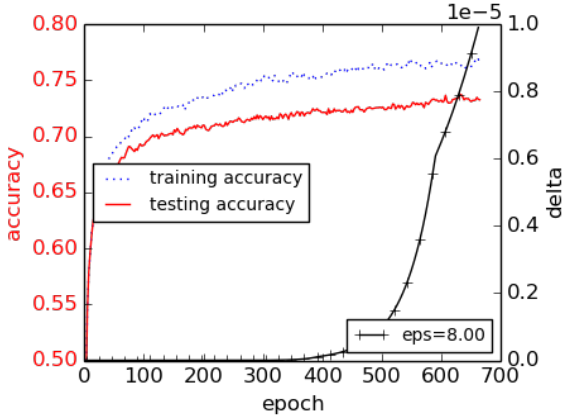
Empirical results on CIFAR10



(1) $\varepsilon = 2$



(2) $\varepsilon = 4$



(3) $\varepsilon = 8$

Figure 6: Results on accuracy for different noise levels on CIFAR-10. With δ set to 10^{-5}, we achieve accuracy 67%, 70%, and 73%, with ε being 2, 4, and 8, respectively. The first graph uses a lot size of 2,000, (2) and (3) use a lot size of 4,000. In all cases, σ is set to 6, and clipping is set to 3.

Abadi et al. (2016) Deep Learning with Differential Privacy: https://arxiv.org/pdf/1607.00133.pdf

Per-Example Clipped Gradient Descent may not converge

- Example: Two data point / linear losses

Knowledge transfer model via the “Private Aggregation of Teacher Ensemble” (PATE)

- Assume a large private dataset
- A small public dataset (can be unlabeled)

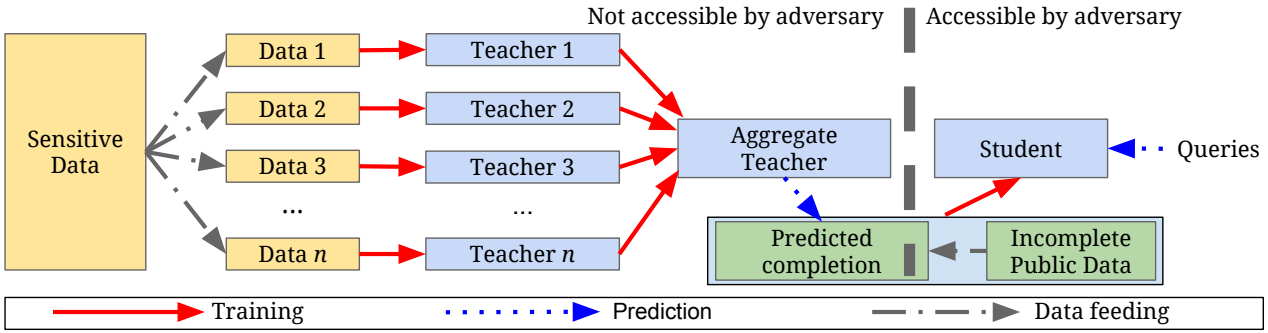


Figure 2: Overview of the approach: (1) an ensemble of teachers is trained on disjoint subsets of the sensitive data, (2) a student model is trained on public data labeled using the ensemble.

Papernot et al. (2018) Scalable private learning with PATE: https://arxiv.org/abs/1802.08908

What does PATE look like from autodp?

- Just a composition of Gaussian mechanisms!

```python
from autodp.mechanism_zoo import GaussianMechanism

import numpy as np

class PATE(GaussianMechanism):
    def __init__(self,sigma, m, Binary,name='PATE'):
        # sigma is the std of the Gaussian noise added to the voting scores
        if Binary:
            # This is a binary classification task
            sensitivity = 1
        else: # for the multiclass case, the L2 sensitivity is sqrt(2)
            sensitivity = np.sqrt(2)
        GaussianMechanism.__init__(self, sigma=sigma/sensitivity/np.sqrt(m),name=name)

        self.params = {'sigma':sigma}
```

- Ideas / tricks that improve to PATE
  - Noisy-Screening and semi-supervised learning (Papernot et al 2017/2018)
  - Release only the argmax rather than voting scores? (Papernot et al 2018)
  - Sparse Vector Technique (Bassily et al., 2018; Liu et al. 2020)
  - Active learning (Liu et al. 2020)

Private K-Nearest Neighbor

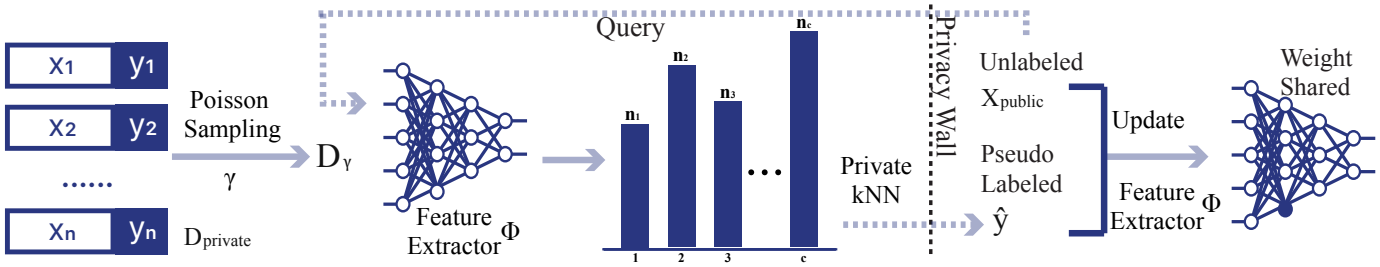


Figure 2. The overview of the proposed framework. Given the unlabeled public data X_{public}, we query through privacy wall for pseudo labels, where the private data and the queried public data are sent through feature extractor Φ and "Private-kNN" to assign pseudo labels. Combining the public data and the pseudo labels, the feature extractor Φ is further updated. This procedure can be iterated for rounds to achieve satisfied privacy-accuracy trade-off.

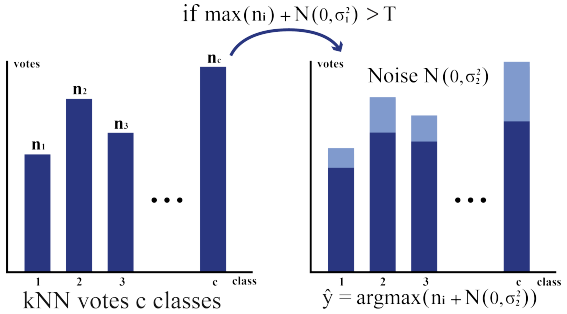


Figure 3. Illustration on the noisy screening and noisy aggregation procedure.

Zhu, Y., Yu, X., Chandraker, M., & Wang, Y. X. (2020). Private-KNN: Practical differential privacy for computer vision. In *CVPR'2020*

What does private KNN look like from autodp?

- Just a composition of Subsampled Gaussian mechanisms (same as NoisySGD!)

```
In [4]: # We will reuse the noisySGD class

        # number of data points

        gamma = 0.01
        m=1000
        sigma = 5.0

        privateKNN = NoisySGD_mech(prob=gamma,sigma=sigma,niter=m)

        # compute epsilon, as a function of delta
        privateKNN.get_approxDP(delta=1e-6)

Out[4]: 0.27105623043762284
```

- Isn't there a NoisyScreening step?

Empirical results in the Knowledge Transfer model

- Private-KNN and PATE:

Table 1. Utility and privacy of semi-supervised student model

Dataset	Methods	#Queries	ϵ	Acc.	NP Acc.
MNIST	LNMAX	1000	8.03	98.1%	99.2%
	GNMAX	286	1.97	98.5%	
	Ours	735	**0.47**	98.8%	
SVHN	LNMAX	1000	8.19	90.1%	92.8%
	GNMAX	3098	4.96	91.6%	
	Ours	2939	**0.49**	91.6%	
CIFAR-10	GNMAX			$\leq$ 50%	80.5%
	Noisy SGD		4	70%	
	Ours	3877	**2.92**	70.8%	

Theory behind PATE and algorithmic improvements

Table 1: Summary of our results: excess risk bounds for PATE algorithms.

Algorithm	**PATE** (Gaussian Mech.) Papernot et al. [2017]	**PATE** (SVT-based) Bassily et al. [2018a]	**PATE** (SVT-based) This paper	**PATE** (Active Learning) This paper
Realizable	$\tilde{O}\Big(\frac{d}{(n\epsilon)^{2/3}} \vee \frac{d}{m}\Big)$	$\tilde{O}\Big(\frac{d}{(n\epsilon)^{2/3}} \vee \sqrt{\frac{d}{m}}\Big)$	$\tilde{O}\Big(\frac{d^{3/2}}{n\epsilon} \vee \frac{d}{m}\Big)$	$\tilde{O}\Big(\frac{d^{3/2}\theta^{1/2}}{n\epsilon} \vee \frac{d}{m}\Big)$
τ-TNC	$\tilde{O}\Big(\big(\frac{d^{3/2}}{n\epsilon}\big)^{\frac{2\tau}{4-\tau}} \vee \frac{d}{m}\Big)$	same as agnostic	$\tilde{O}\Big(\big(\frac{d^{3/2}}{n\epsilon}\big)^{\frac{\tau}{2-\tau}} \vee \frac{d}{m}\Big)$	$\tilde{O}\Big(\big(\frac{d^{3/2}\theta^{1/2}}{n\epsilon}\big)^{\frac{\tau}{2-\tau}} \vee \frac{d}{m}\Big)$
Agnostic (vs h^*)	$\Omega(\mathtt{Err}(h^*))$ required.	$13\mathtt{Err}(h^*)+$ $\tilde{O}\Big(\frac{d^{3/5}}{n^{2/5}\epsilon^{2/5}} \vee \sqrt{\frac{d}{m}}\Big)$	$\Omega(\mathtt{Err}(h^*))$ required.	$\Omega(\mathtt{Err}(h^*))$ required.
Agnostic (vs $h_\infty^{\mathtt{agg}}$)	-	-	Consistent under weaker conditions.	-

- What do these results say:
  - Sparse-Vector-Technique-based online query release helps
  - Active learning helps
  - Label-noise models help

Liu, Zhu, Chaudhuri and W. (2020) "Revisiting model-agnostic private learning". AISTATS and JMLR. https://arxiv.org/pdf/2011.03186.pdf

Next lecture

- Going beyond the worst case!
- Smoothed Sensitivity and the Median