Chike Abuah
Alex Silence
David Darais
Joe Near

UVM PLAID
Galois Inc.

DDUO: General-Purpose Dynamic Analysis for Differential Privacy

DDUO makes it easy to automatically enforce privacy

```python
def dp_gradient_descent(iterations, alpha, epsilon):
  eps_i = epsilon/iterations
  theta = np.zeros(X_train.shape[1])
  for i in range(iterations):
    grad_sum        = gradient_sum(theta, X_train, y_train, sensitivity)
    noisy_grad_sum  = duet.renyi_gauss_vec(grad_sum, α=alpha, ε=eps_i)
    noisy_avg_grad  = noisy_grad_sum / X_train.shape[0]
    theta           = theta - noisy_avg_grad
  return theta
```

- Usable
- For non-experts
- Capable of complex algorithms

DDUO is prototyped in Python

- DDUO provides a dynamic analysis for enforcing differential privacy, embedded in a general purpose language (Python).
- Expresses privacy concepts such as sequential composition naturally in Python with pythonic idioms such as with blocks.

```
with dduo.EpsOdometer() as odo:
  _ = dduo.laplace(df.shape[0], ε = 1.0)
  _ = dduo.laplace(df.shape[0], ε = 1.0)
  print(odo)
```

The DDUO Analysis System

Sensitivity Analysis: Object Proxies

```python
from dduo import pandas as pd
df = pd.read_csv("data.csv")
df

# no change to sensitivity environment
df + 5

# doubles the sensitivity
df + df

( df * 5, df * df)
```

>>> *Sensitive(<'DataFrame'>, data.csv* $\mapsto$ *1, L∞)*

>>> *Sensitive(<'DataFrame'>, data.csv* $\mapsto$ *1, L∞)*

>>> *Sensitive(<'DataFrame'>, data.csv* $\mapsto$ *2, L∞)*

>>> *(Sensitive(<'DataFrame'>, data.csv* $\mapsto$ *5, L∞), Sensitive(<'DataFrame'>, data.csv* $\mapsto$ *∞, L∞))*

Conditionals

```
if df.shape[0] == 10:
  return df.shape[0]
else:
  return df.shape[0] * 10000

if dduo.gauss(ε=1.0, δ=1e-5, x) > 5:
  print(dduo.gauss(ε=1.0, δ=1e-5, y))
else:
  print(dduo.gauss(ε=100000000000.0, δ=1e-5, y))
```

- Branching on sensitive values is disallowed

- Adaptive privacy analysis requires use of odometers/filters

Privacy Analysis: Filters and Odometers

```
dduo.laplace(df.shape[0], ε=1.0)

with dduo.EpsOdometer() as odo:
  _ = dduo.laplace(df.shape[0], ε = 1.0)
  _ = dduo.laplace(df.shape[0], ε = 1.0)
  print(odo)

with dduo.EdFilter(ε = 1.0, δ = 10e-6) as odo:
  print('1:', dduo.gauss(df.shape[0], ε=1.0, δ=10e-6))
  print('2:', dduo.gauss(df.shape[0], ε=1.0, δ=10e-6))
```

>>> 9.963971319623278

>>> Odometer_ε(data.csv ↦ 2.0)

>>> 1: 10.5627

Traceback (most recent call last):

...

dduo.PrivacyFilterException

Loops, Composition, Variants

```
# sequential composition
with dduo.EpsOdometer() as odo:
  for i in range(20):
    dduo.laplace(df.shape[0], ε = 1.0)
  print(odo)

# advanced composition
with dduo.AdvEdOdometer() as odo:
  for i in range(20):
    dduo.gauss(df.shape[0], ε = 0.01, δ = 0.001)

# variant mixing
with dduo.EdOdometer(max_delta = 1e-4) as odo:
  with dduo.RenyiDP(1e-5):
    for x in range(200):
      noisy_count = dduo.renyi_gauss(α = 10,
        ε=0.2, df.shape[0])
  print(odo)
```

- sequential composition

- advanced composition

- variant mixing

Gradient Descent in DDUO

```
def dp_gradient_descent(iterations, alpha, epsilon):
  eps_i = epsilon/iterations
  theta = np.zeros(X_train.shape[1])
  for i in range(iterations):
    grad_sum        = gradient_sum(theta, X_train, y_train, sensitivity)
    noisy_grad_sum  = duet.renyi_gauss_vec(grad_sum, α=alpha, ε=eps_i)
    noisy_avg_grad  = noisy_grad_sum / X_train.shape[0]
    theta           = theta - noisy_avg_grad
  return theta
```

Case Studies: Dynamic Enforcement of Privacy

Algorithm	Libraries Used	Baseline	Instrumented Version	Overhead (% increase)
Noisy Gradient Descent	NumPy	5.922s	6.302s	6.42%
Multiplicative Weights (MWEM)	Pandas	0.725s	0.833s	14.90%
Private Naive Bayes Classification	DiffPrivLib	2.155s	2.423s	12.44%
Private Logistic Regression	DiffPrivLib	2.022s	3.161s	56.33%

DDUO: General-Purpose Dynamic Analysis for Differential Privacy

- Enforcement for differential privacy is important:
  Buggy programs silently violate your privacy
- Automated enforcement of privacy can be practical

https://github.com/uvm-plaid/dduo-python

Contributors

Chike Abuah, Alex Silence, Vanessa White, David Darais, Joe Near