Reproducing Result of 'DeepMutation: Mutation Testing of Deep Learning Systems'

Prepared by-Sumaiya Saima Sultana

Summary of paper

- **Goal:** Quality evaluation and weakness localization of test dataset
- **Approach**: Incorporating the idea of Mutation Testing in DL systems
- **Claimed contributions**:
  - Designing 8 source level mutation operators & 8 model level mutation operators to introduce faults in systems
  - Introducing two new DL specific mutation testing metrics to allow quantitative measurement of test quality
- **Datasets used for evaluation**: MNIST & CIFAR-10

Terms used:

- **Mutation Testing in DL:** Mutation testing in DL is a method of measuring quality of a test data set by injecting potential faults into training dataset/training program or trained models.
- **Source Level Mutation Operators:** These are the operators to manipulate either training data or training program. They are called source level because mutation is done before training the model.
- **Model Level Mutation Operators:** These are the operators to mutate trained DL models. They directly mutate the structures and parameters of DL models.

Terms used:

- **Mutant models:** The models produced after applying different mutation operators are called mutant models.
- **Mutation score:** It can be defined as the ratio of killed classes by a mutant model to the total classes. (This definition mainly focuses on classification problems)
- **Passed test set:** Passed test set contains only those data points of the original test dataset which were truly classified by the original model.
- **Average error rate:** This is the average rate of error for mutant models. This metric is used to control the quality of mutant model itself. If ER of a mutant model is very high, then the model should be excluded from the testing since it introduces large behavioral difference.

Mutation Testing Evaluation & Results

- **Workflow for Source Level Mutation Testing using Uniform Sampling**

Load data and extract 5000 datapoints from training dataset and 1000 from test dataset

Check if the data is uniform

Apply required **preprocessing** (i.e. reshaping, standardizing,one hot encoding etc.)

Create a model(A/B/C) and train the model using training data and report accuracy

Make predictions on test data and create filtered dataset(T') by extracting only correctly predicted datapoints

Create data frame to save results of mutation score and accuracy on T'

For each mutation operator:

- Mutate traindata/train program
- Train mutant/original program using original/mutant data to create mutant model m'
- Make predictions on T' using m'
- Calculate Mutation score,accuracy on T'

Calculate Average Mutation Score for SL mutation and AER

Mutation Testing Evaluation & Results

- **Workflow for Source Level Mutation Testing using Non-Uniform Sampling**

Specify distribution ratio, class to prioritize and seed value.

Create two subsets of each dataset, one for chosen class & one for other classes.

Randomly choose (20-50%) data from first set and left portion from other set to create non uniform dataset.

Print number of data in each class to check if the data is uniform

Apply required **preprocessing** (i.e. reshaping, standardizing, one hot encoding etc.)

Create a model(A/B/C) and train the model using training data and report accuracy

Make predictions on test data and create filtered dataset(T') by extracting only correctly predicted datapoints

Create data frame to save results of mutation score and accuracy on T'

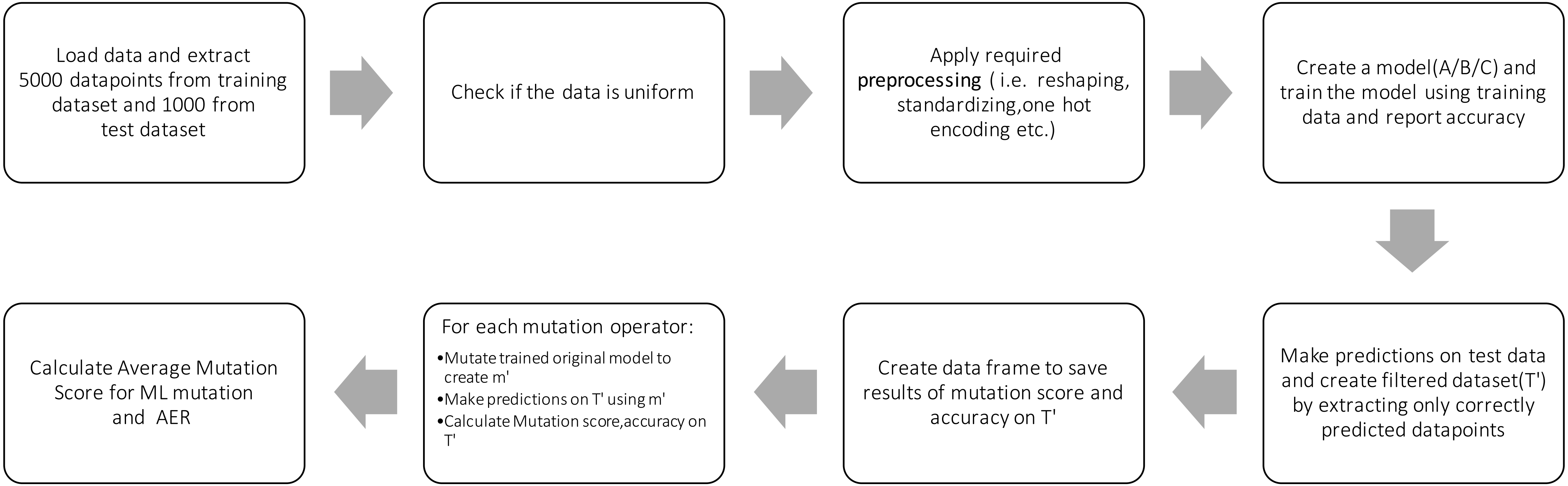
For each mutation operator:

- Mutate traindata/train program
- Train mutant/original program using original/mutant data to create mutant model m'
- Make predictions on T' using m'
- Calculate Mutation score, accuracy on T'

Calculate Average Mutation Score for SL mutation and AER

Mutation Testing Evaluation & Results

- **Workflow for model Level Mutation Testing using Uniform Sampling**



Mutation Testing Evaluation & Results

- **Workflow for Model Level Mutation Testing using Non-Uniform Sampling**

Specify distribution ratio, class to prioritize and seed value.

Create two subsets of each dataset, one for chosen class & one for other classes.

Randomly choose (20-50%) data from first set and left portion from other set to create non uniform dataset.

Print number of data in each class to check if the data is uniform

Apply required **preprocessing** (i.e. reshaping, standardizing,one hot encoding etc.)

Create a model(A/B/C) and train the model using training data and report accuracy

Make predictions on test data and create filtered dataset(T') by extracting only correctly predicted datapoints

Create data frame to save results of mutation score and accuracy on T'

For each mutation operator:

- Mutate trained original model to create m'
- Make predictions on T' using m'
- Calculate Mutation score,accuracy on T'

Calculate Average Mutation Score for SL mutation and AER

Summary of Results

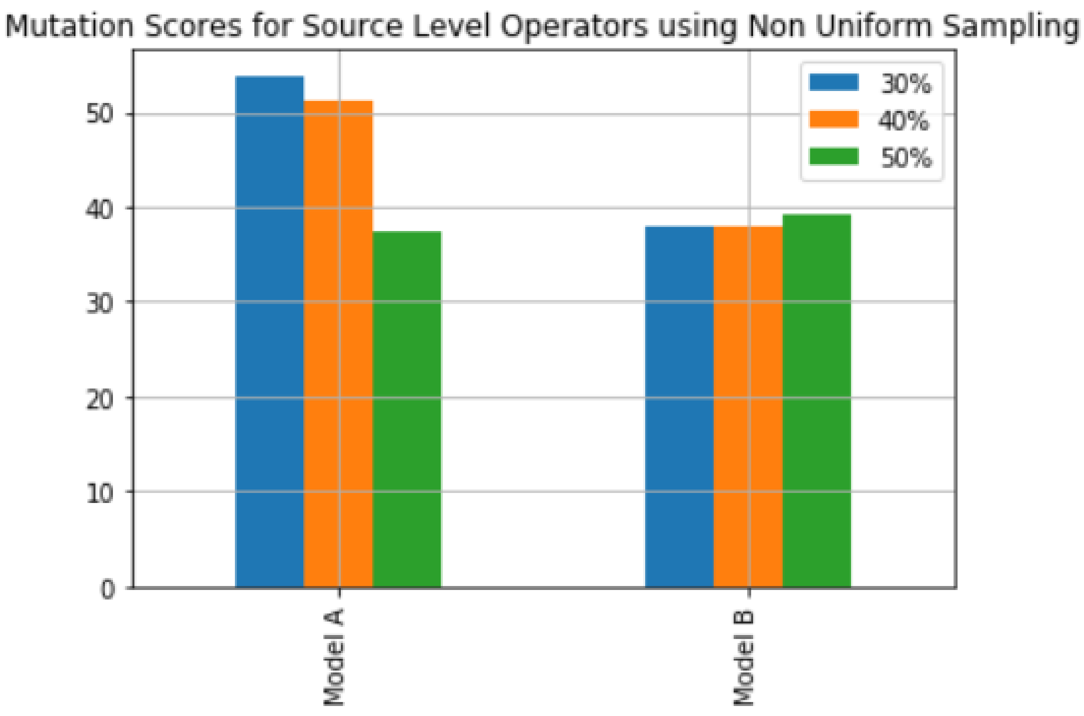


Figure: Mutation scores(%) of SL mutant models using various non uniform sampling (For class label = 2)

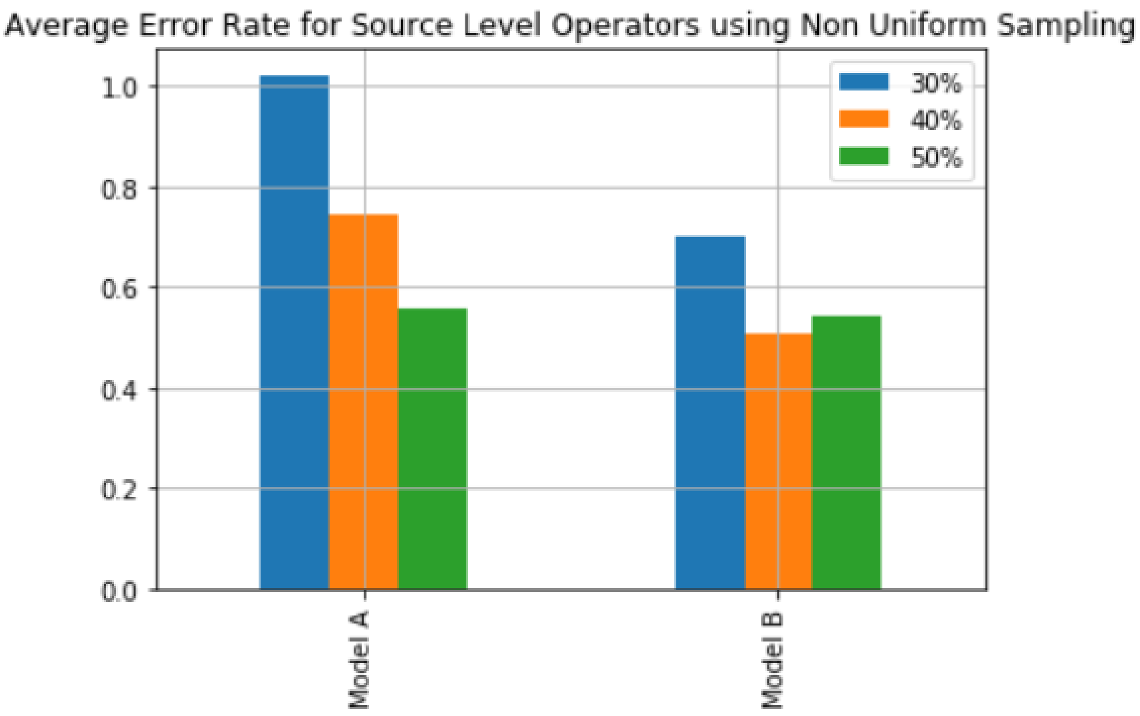


Figure: Average Error Rate (%) of SL mutant models using various non uniform sampling (For class label = 2)

Summary of Results

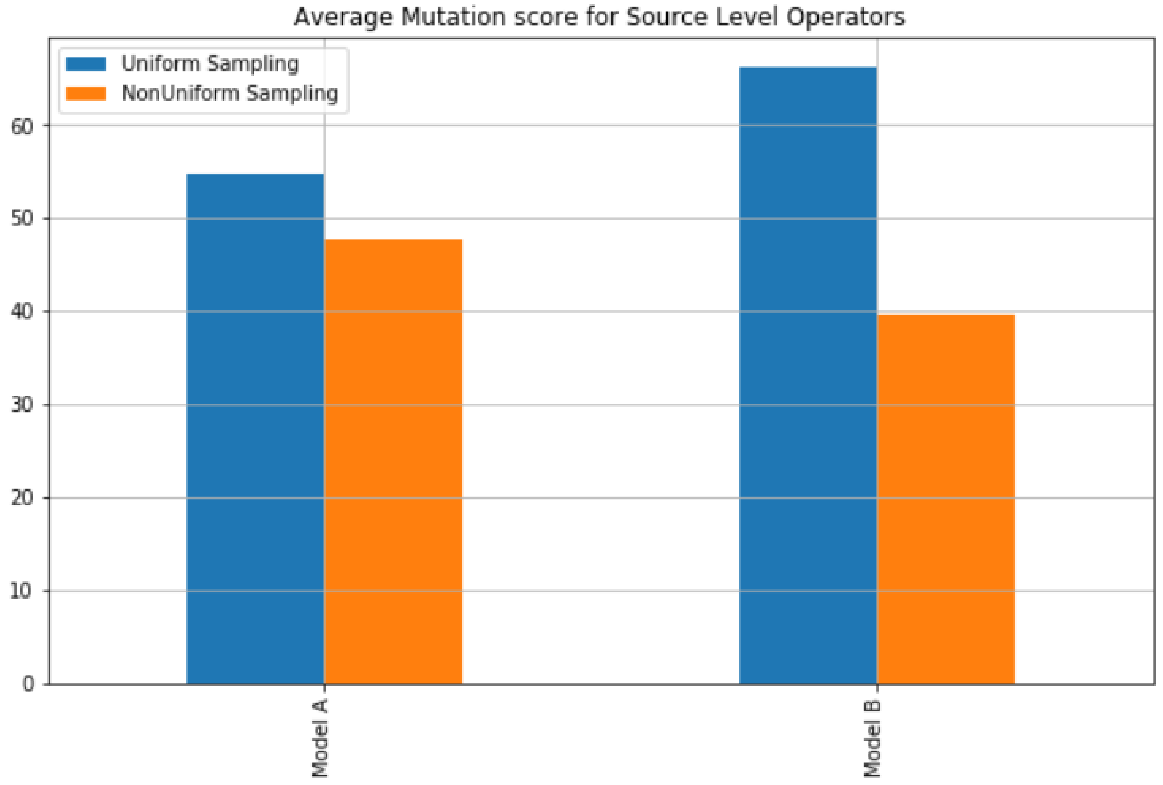


Figure: Average Mutation score (%) of SL mutant models

	Uniform Sampling	NonUniform Sampling
Model A	54.750	47.541667
Model B	66.125	39.500000

Summary of Results

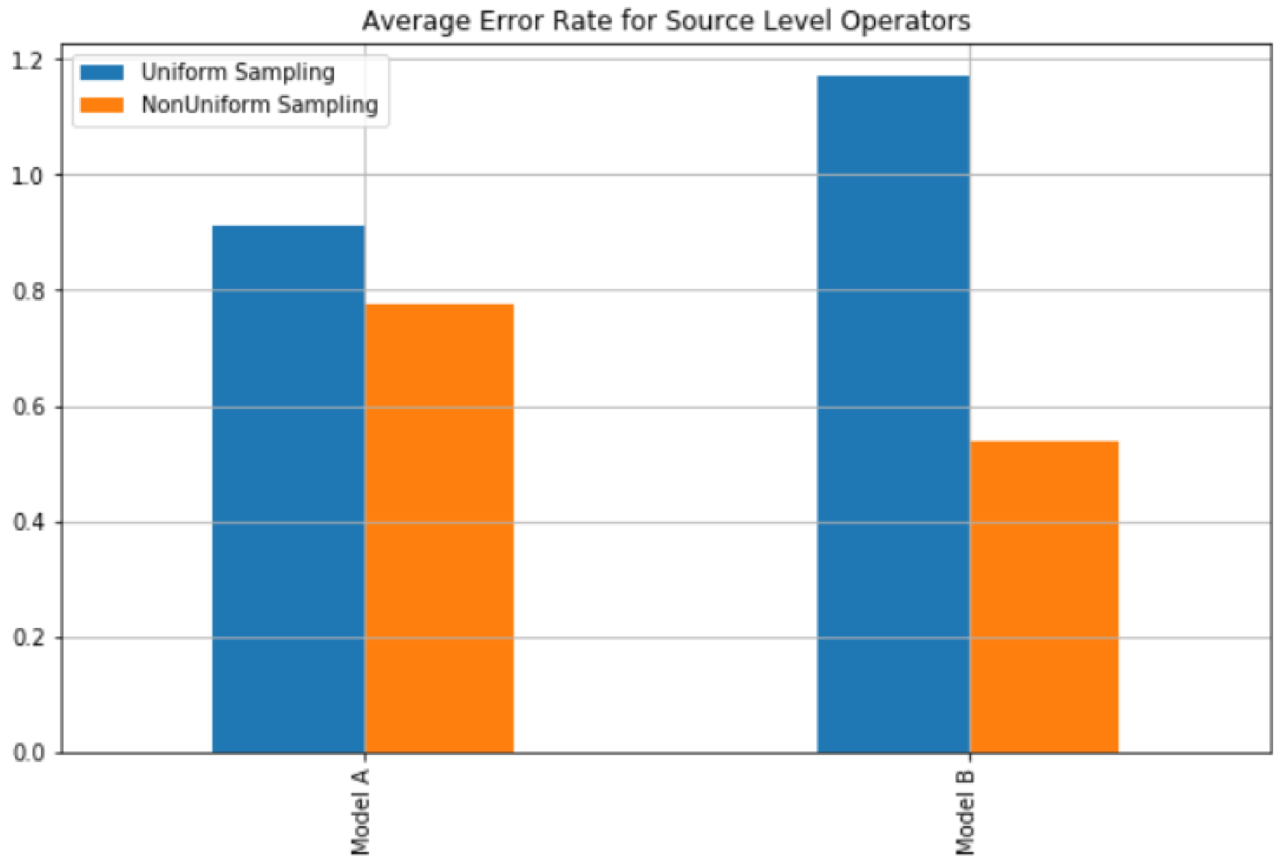


Figure: Average Error Rate (%) of SL mutant models

	Uniform Sampling	NonUniform Sampling
Model A	0.909	0.775000
Model B	1.170	0.539333

Comments on results:

- Produced results of Model A and B matches the pattern of original results of the paper
- Non-Uniform testing has lower mutation score than Uniform testing as expected.
- As non uniformity increases, mutation score seems to decrease.

Thank you‼