TestRank: Bringing Order into Unlabeled Test Instances for Deep Learning Tasks

Yu LI, Min Li, Qiuxia Lai, Yannan Liu*, and Qiang Xu

Reliable Computing Lab. (CURE), The Chinese University of Hong Kong

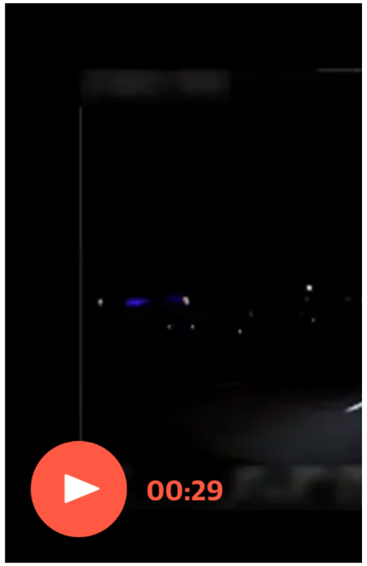
*Wuheng Lab, ByteDance

CUhk REliable Computing Laboratory

Uber crash shows 'catastrophic failure' of self-driving technology, experts say

Concerns raised about futu
collision in Arizona was fail

- **Video released of fatal Ub**



▲ Uber dashcam footage shows lead up t

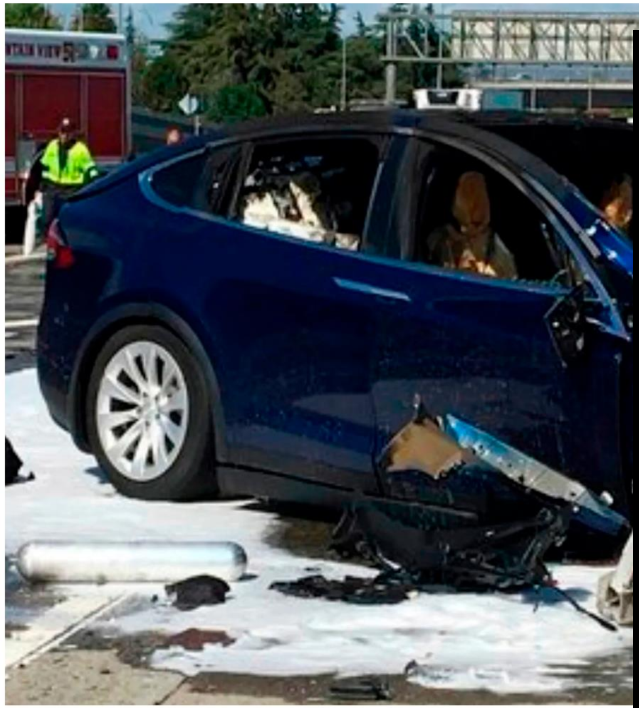
Video of the first self-drivin
"catastrophic failure" by Ub
who said the footage showe
most basic functions.

Tesla needs to fix its deadly Autopilot problem

Tesla is facing heat from federal officials following an investigation into a fatal crash involving its Autopilot.

By Rebecca Heilweil | Feb 26, 2020, 1:50pm EST

SHARE



Wrongfully Accused by an Algorithm

In what may be the first known case of its kind, a faulty facial recognition match led to a Michigan man's arrest for a crime he did not commit.

Why AI Systems Fail?

- Inproper Training
  - Insufficient/Dirty/Maliciously injected training data
  - Weak model structure
  - Insufficient training epochs

(a) Input 1 (b) Input 2 (darker version of 1)

Nvidia DAVE-2 self-driving car platform
A failure caused by the darkness [1]

1.1 original 1.2 with added rain

A failure caused by the rain
in the Chauffeur DNN [2]

Hence, testing of AI-based systems is important before deployment

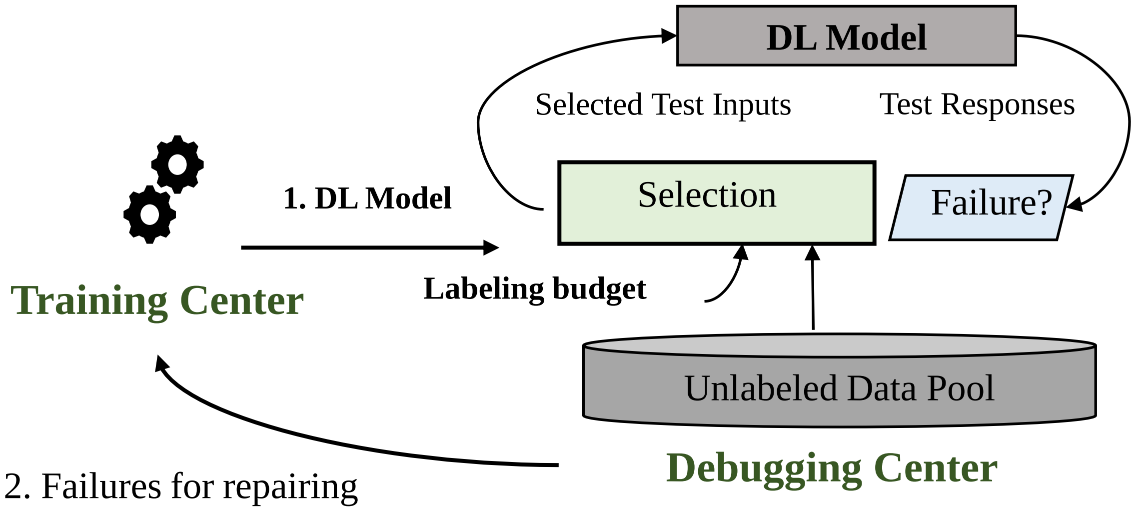
[1] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. 2019. DeepXplore: automated whitebox testing of deep learning systems. <i>Commun. ACM</i> 62, 11 (November 2019), 137–145. DOI:https://doi.org/10.1145/3361566

[2] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. DeepTest: automated testing of deep-neural-network-driven autonomous cars. In <i>Proceedings of the 40th International Conference on Software Engineering</i> (<i>ICSE '18</i>). Association for Computing Machinery, New York, NY, USA, 303–314. DOI:https://doi.org/10.1145/3180155.3180220

Test Sample Prioritization and Selection

- DL system is data driven
- Massive unlabeled test instances
- Limited labeling resources

The test prioritization problem:
Given a large amount of unlabeled test data and certain labeling budget, how to select test cases that reveals more DNN behavior errors (failures)?



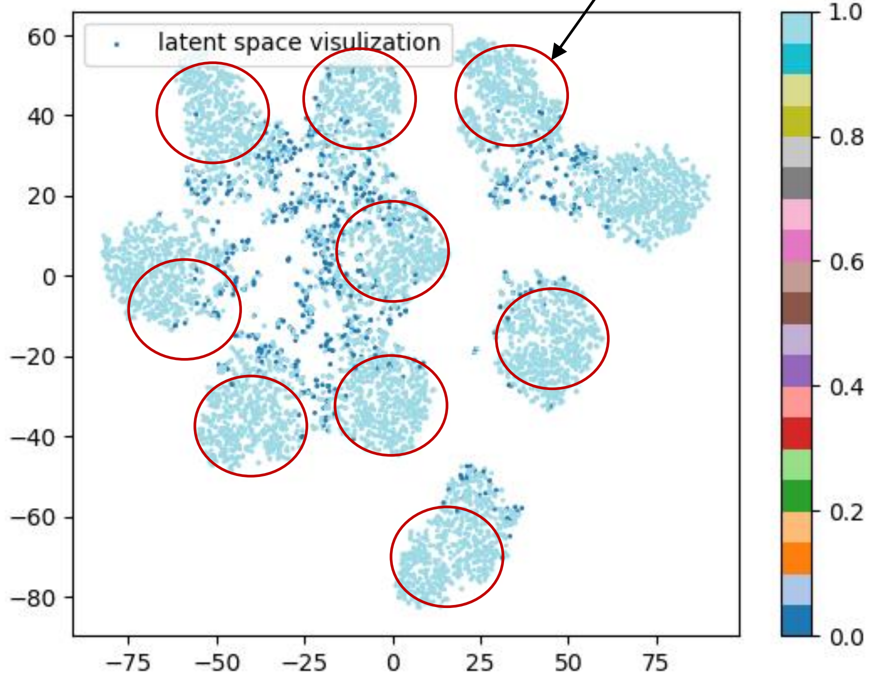
The general testing/debugging overflow.

Select 100 test cases, detect 2 failures
Select 100 test cases, detect 50 failures! ✓

Test Sample Selection – The Problem of Random Selection

- For a well-trained DL classifier, most of the selected samples can be correctly classified

These areas are likely to be selected by random selection



Light Blue: correctly classified ; Dark Blue: misclassification

t-SNE visualization of CIFAR-10 images

Representive Existing Solutions

- Confidence based (DeepGini [1])
  - Confidence score = $\sum p_i^2$
  - Select test cases with low score
  - Example: For output vector [0.1, 0.9] and [0.5, 0.5], they select [0.5, 0.5]

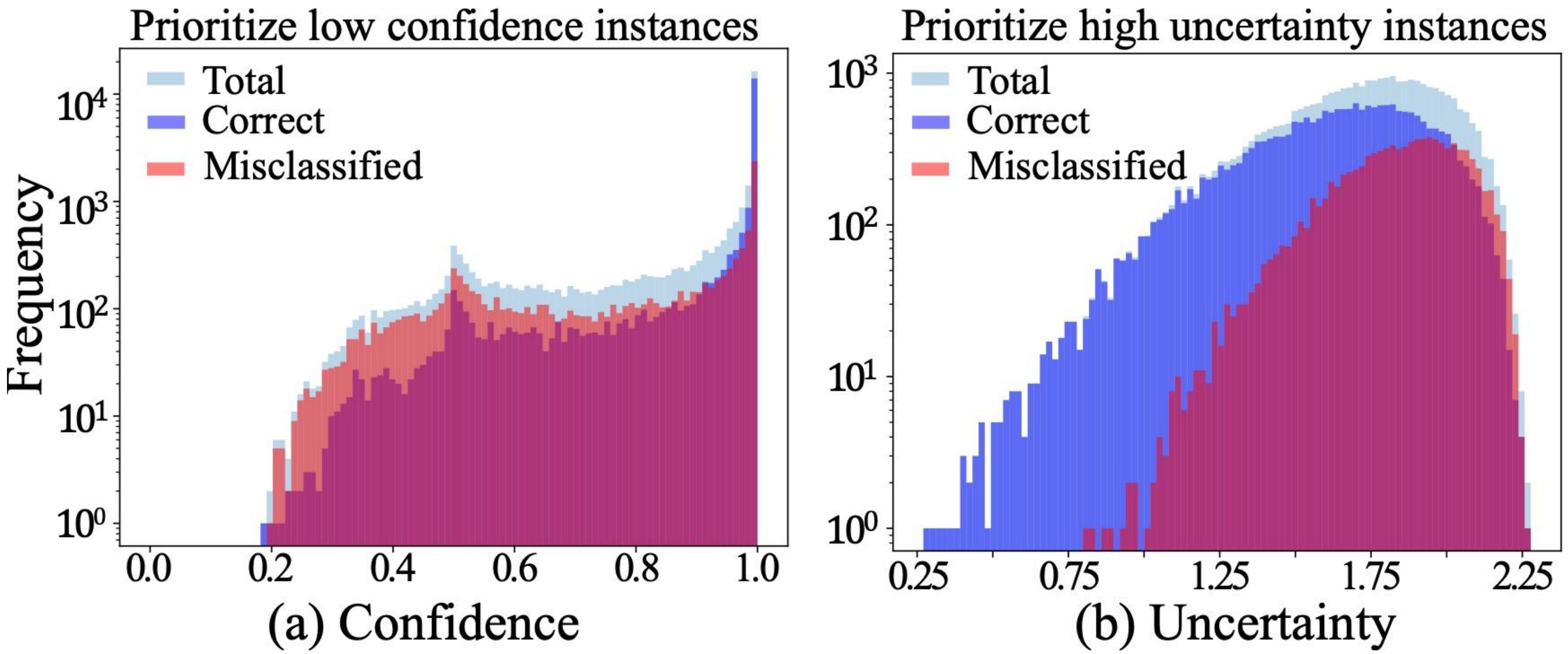
- Bayesian uncertainty based [2]
  - Run the DL model with certain dropout rate *T* times
  - Average the model outputs
  - Calculate the entropy on the averaged output

- MCP [3]
  - Balance confidence and classes among selected test instances

[1] Feng, Y., Shi, Q., Gao, X., Wan, J., Fang, C., & Chen, Z. (2020, July). DeepGini: prioritizing massive tests to enhance the robustness of deep neural networks. In *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis* (pp. 177-188).
[2] Byun, T., Sharma, V., Vijayakumar, A., Rayadurgam, S., & Cofer, D. (2019, April). Input prioritization for testing neural networks. In *2019 IEEE International Conference On Artificial Intelligence Testing (AITest)* (pp. 63-70). IEEE.
[3] Shen, W., Li, Y., Chen, L., Han, Y., Zhou, Y., & Xu, B. (2020, September). Multiple-Boundary Clustering and Prioritization to Promote Neural Network Retraining. In *2020 35th IEEE/ACM International Conference on Automated Software Engineering (ASE)* (pp. 410-422). IEEE.

The Problem of Existing Solutions

Histogram of confidence and uncertainty of a CIFAR-10 model



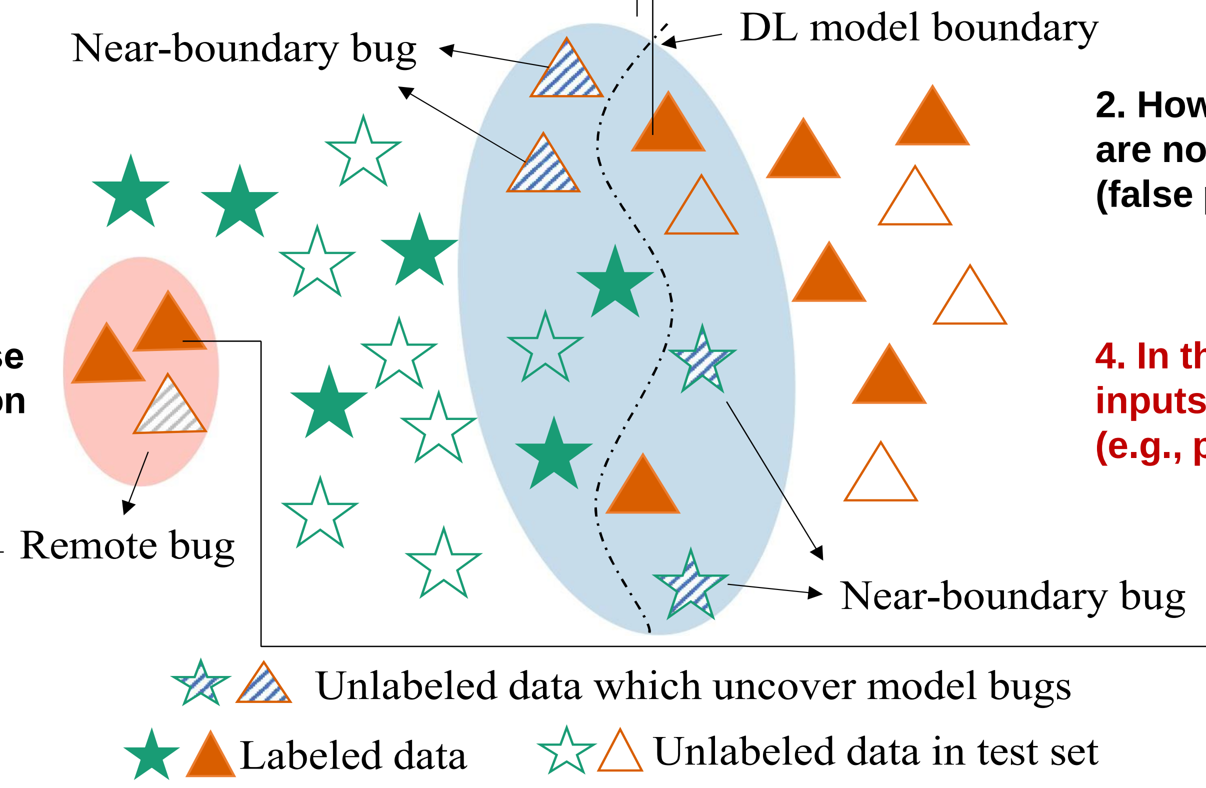
Observation

- Low confidence/High uncertainty does not mean misclassification
- Misclassifications can have high confidence/low uncertainty

Motivational Example

1. Existing approaches aim to select near-boundary instances

2. However, near-boundary instances are not necessarily been misclassified (false positives)



3. What's worse, they ignore those bugs that are far from the decision boundary, i.e., remote failures (false negatives)

4. In the meantime, the historical test inputs and responses are informative (e.g., previously labeled test instances)

If we make use of these contextual information, we can detect both near-boundary and remote failures

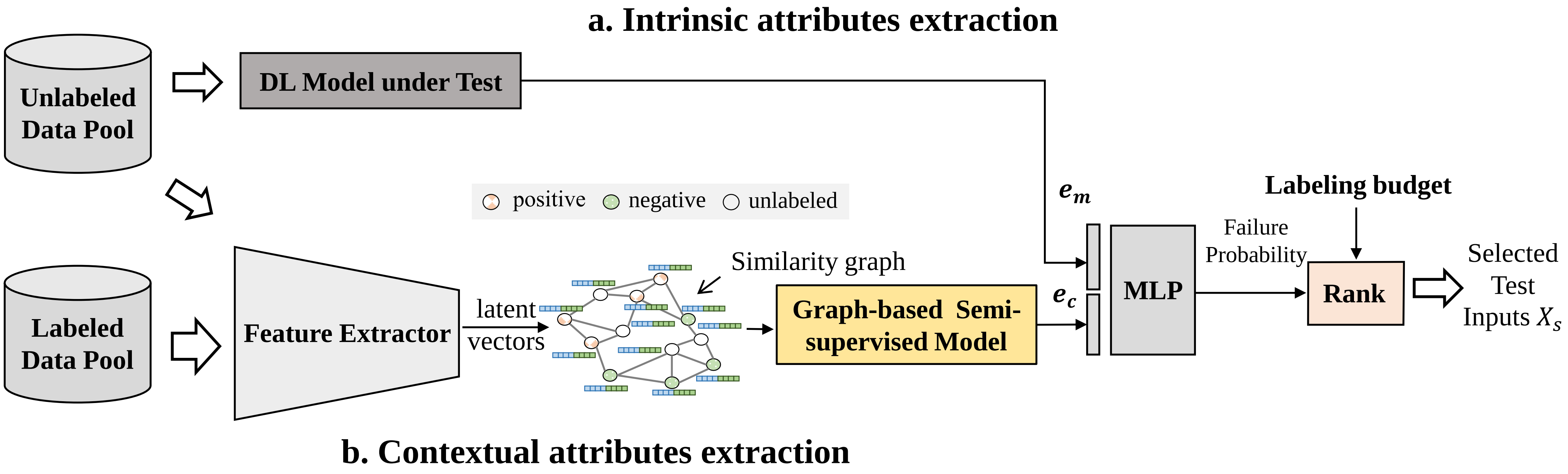
Core Idea of Our Solution –TestRank

TestRank make use of **both** Intrinsic **and** contextual attributes

- **Intrinsic attributes**
  - The output vectors from the DL model
  - Though not accurate, but a still useful indicator of near-boundary failures

- **Contextual attributes**
  - **Summarized correctness from the neighboring labeled samples**
    - E.g., Most labeled neighbors are misclassified samples
  - Help intrinsic attributes to reduce false positives and false negatives

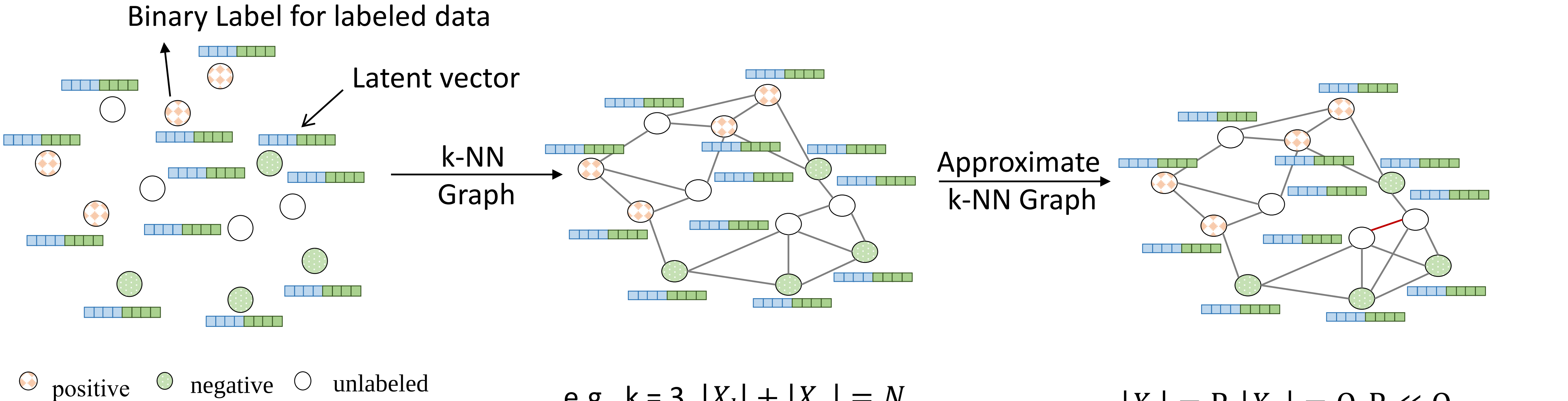
The Overflow of TestRank



- Combination of intrinsic (a) and contextual attributes (b) for failure probability estimation
- Graph Neural Networks (GNN) is good at extracting contextual features

Graph Construction

- k-nearest neighbor (k-NN) graph: **connecting the nearest k neighbors**
- **The connections between unlabeled data are less important**
- Approximate k-NN graph:
  - **only connect unlabeled data with labeled data, and labeled data to labeled data**

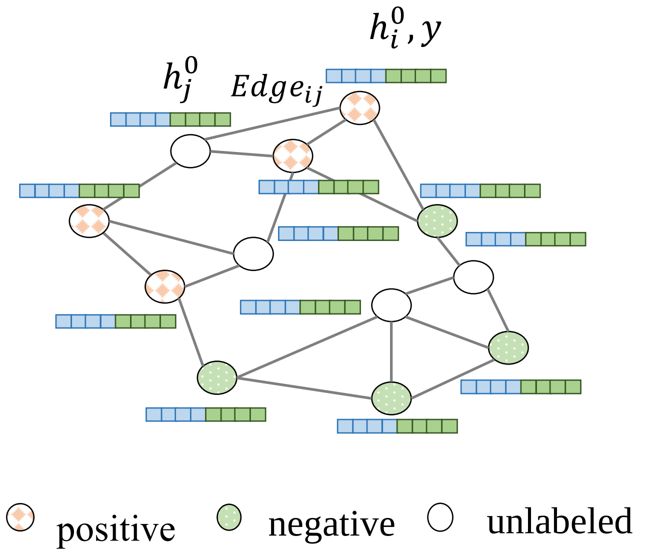


e.g., k = 3, $|X_l| + |X_u| = N$
Computational complexity: $O(N^2)$

$|X_l| = \mathrm{P}, |X_u| = \mathrm{Q}, \mathrm{P} \ll \mathrm{Q}$
Computational complexity: $O(PQ)$

Graph Neural Network for Contextual Attributes Extraction

- Apply semi-supervised GNN on the similarity graph $\mathrm{G}(\mathrm{H}, Edge)$
- A GCN layer: $H_{i+1} = \underbrace{\alpha}_{\text{Activate}}(\widehat{D}^{-\frac{1}{2}} \underbrace{\hat{A}}_{\text{Aggregate}} \widehat{D}^{-\frac{1}{2}} H_i \underbrace{W}_{\text{Transform}})$



/* KNN Graph construction */

$\mathrm{A}, \mathrm{Edge} = knn\_graph(\overline{\mathbf{X}}, k)$;

/* Train GNN */

$\tilde{\mathbf{A}} = \mathbf{Edge} + \mathbf{I}_N$;

$\tilde{\mathbf{D}} = \sum_j \tilde{\mathbf{A}}_{i,j}$;

$\mathbf{H}^0 = \overline{\mathbf{X}}$;

for *Number of training epochs* **do**

  for $l = 0, 1, \ldots, M$ **do** → M GNN layers

    $\mathbf{H}^{l+1} = \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{H}^l \Theta^l)$, → Aggregate information from neighbors

  end

  $Output = \mathrm{FCLayer}(\mathbf{H}^{M+1})$;

  $loss = \mathrm{CrossEntropyLoss}(Output, \mathbf{Y}_L)$; → Train GNN with CE loss

  Back propagation;

  Update $\mathbf{\Theta}$;

end

$\mathbf{E}_c = \mathbf{H}^{M+1}[\mathrm{unlabeled\_index}]$; → Extract the contextual attributes

Comparison of TestRank with Baseline Methods

- Metric

$$TRC = \frac{\#\, Detected\ Bugs}{\min(\, Budget,\ \ \#\, Total\ bugs)}$$

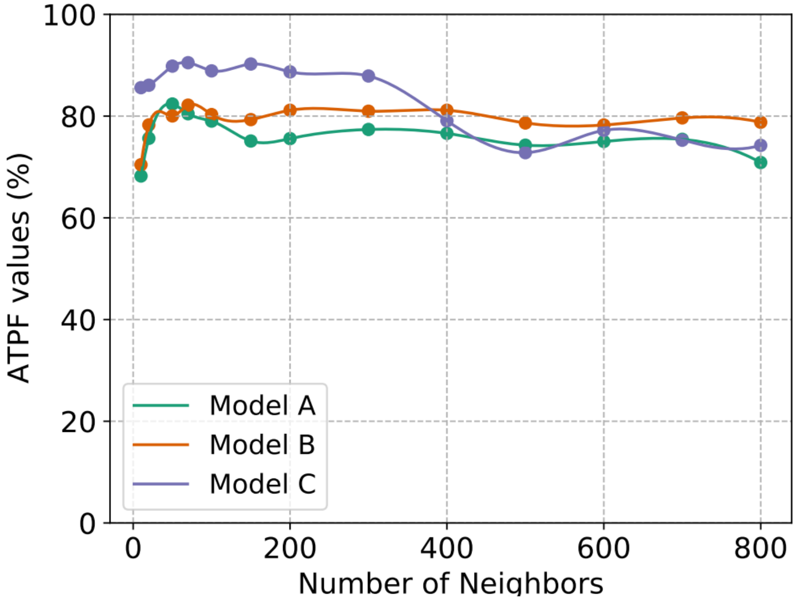
- The table shows the **average TRC** calculated for budget less than the number of total bugs

Dataset	Model ID	Random	MCP	DSA	Uncertainty	DeepGini	TestRank Contextual-Only	TestRank
CIFAR-10	A	30.15	58.25	60.93	58.09	67.47	51.39	**76.56**
	B	34.18	46.46	62.34	61.85	67.80	58.85	**87.87**
	C	34.27	65.25	64.47	63.10	71.15	75.33	**85.53**
SVHN	A	10.16	39.98	55.47	58.29	63.47	44.16	**66.06**
	B	11.85	38.07	57.96	58.06	63.85	51.26	**76.36**
	C	23.41	65.33	69.34	71.80	81.68	93.99	**95.32**
STL10	A	39.25	66.62	64.56	64.30	69.70	60.09	**79.00**
	B	42.60	69.97	67.12	65.30	72.89	71.90	**80.96**
	C	46.05	71.88	66.60	70.34	73.34	79.55	**88.67**

- The contextual information is useful to improve test prioritization effectiveness
- The context attributes alone are not sufficient
- The combination of intrinsic and contextual attributes outperfroms other methods for a large margin

Ablation Study

Dataset	Model	TestRank (%)	TestRank w/o approx. (%)
CIFAR-10	A	76.56	77.77 (+1.21)
	B	87.87	87.70 (-0.17)
	C	85.53	88.10 (+2.57)
SVHN	A	66.06	63.87 (-2.19)
	B	76.36	82.04 (+5.68)
	C	95.32	96.62 (+1.30)
STL10	A	79.00	80.50 (+1.50)
	B	80.96	78.98 (-1.98)
	C	88.67	89.32 (+0.65)
Average Influence (%)			**+0.95**



The influence of approximated kNN construction

The impact of the number of neighbors k on the debug effectiveness (STL10 dataset)

The average influence of the approximation is 0.95%, which is small.

TextRank can achieve good performance in a wide range of k values.

Conclusion

- We propose *TestRank*, a novel test prioritization framework for DL systems

- *TestRank* not only leverages the intrinsic attributes of an input instance, but also extracts the contextual attributes from the DL model's historical inputs and responses

- TestRank constantly outperform other test prioritization methods

Thanks for Listening !

Q & A