Exact Robustness Certification of *k*-Nearest Neighbor Classifiers

Ahmad Shakeel

Master Degree in Computer Science

Università degli Studi di Padova

10 Apr 2025

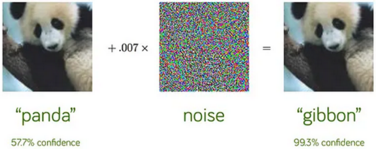
Università degli Studi di Padova

Outline

Adversarial Examples

Definition

Adversarial examples are carefully crafted input data designed to cause an AI system to produce incorrect or biased predictions (Szegedy et al., 2014).



Source: Szegedy et al. Explaining and harnessing adversarial examples

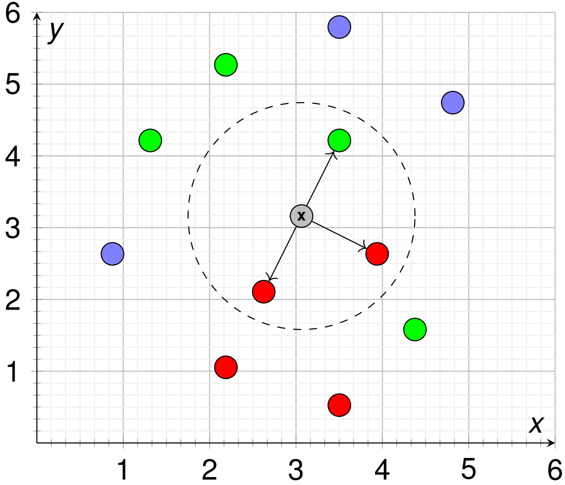
Why do we care ?

- Adversarial examples pose a significant security risks in safety-critical domains (e.g., healthcare, autonomous transportation, etc.)
  - Compromised security systems;
  - Data breaches and unauthorized access;
  - Financial losses and fraud;
  - ...

k-NN

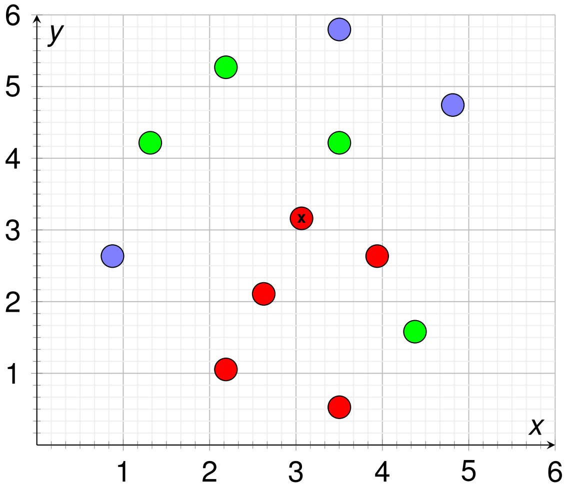
- A *non-parametric* supervised machine learning method;
- Suitable for both classification and regression tasks;
- Leverages data similarity to compute the prediction;
- Relies on distance metrics like Euclidean Manhattan or the general Minkowski distance;

3-NN



3NN over a dataset with 3 classes, which shows how a new point is classified

3-NN



3NN over a dataset with 3 classes, which shows how a new point is classified

Goal

- Exact robustness certification of the k-Nearest Neighbors Classifier;
- Focus on *completeness*;
- Consider only the Euclidean distance as similarity metric.

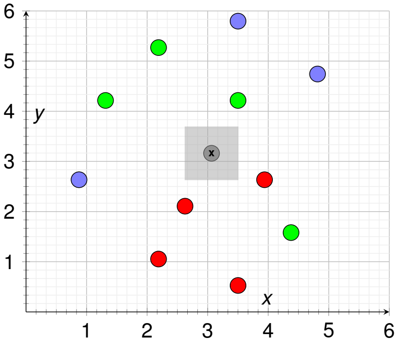
Robustness Certification: Basic idea

- The possible perturbations of x create a small ℓ_∞ ball centered in x with radius $\epsilon > 0$:

$$P^\epsilon(x) = \{x' \in \mathbb{R}^n \mid ||x' - x||_\infty \leq \epsilon\}$$

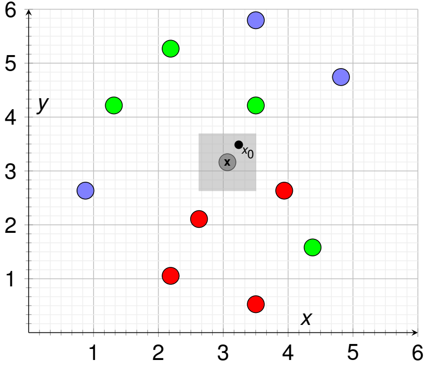
x

- $P^\epsilon(x)$ is called the *Perturbation (or Adversarial)* region of x



Robustness Certification: Stability

- Check if exists $x_0 \in P^{\epsilon}(x)$ such that k-NN$(x) \neq k$-NN(x_0)
- x_0 exists $\Rightarrow$ k-NN is **not stable** on x
- x_0 does not exist $\Rightarrow$ k-NN is **stable** on x



Algorithm overview

Given an input samples x:

Algorithm overview

Given an input samples x:

1. Build a directed graph G where
   - Nodes are samples in the training set;
   - Edges model the relation of ***being-closer*** to the adversarial region of x;

Algorithm overview

Given an input samples x:

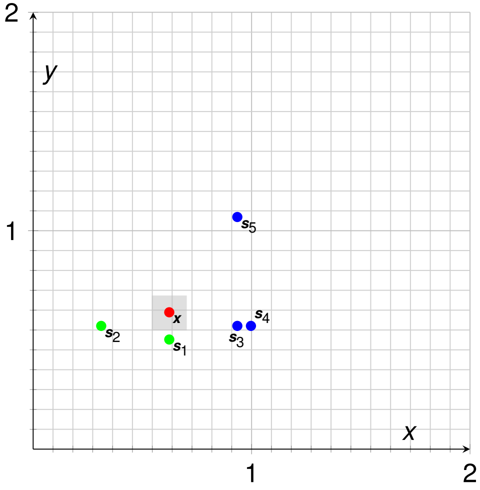
1. Build a directed graph G where
    - Nodes are samples in the training set;
    - Edges model the relation of ***being-closer*** to the adversarial region of x;
2. For each label ℓ
    - Traverse the graph G
    - Find a ***valid*** path with k samples where ℓ is dominant label;

Algorithm overview

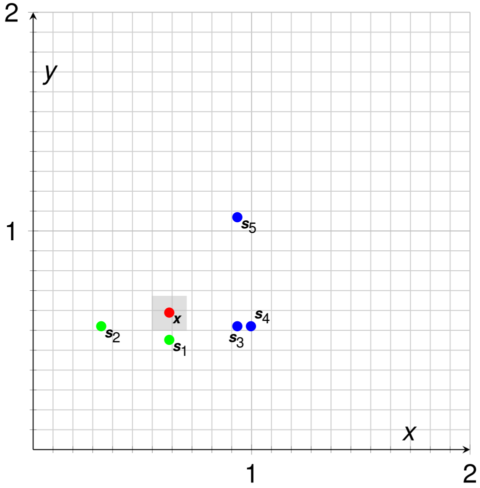
Given an input samples x:

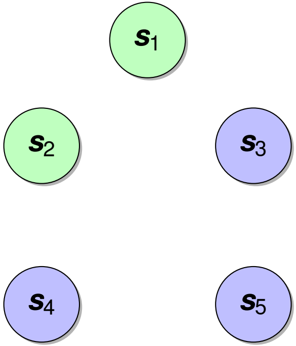
1. Build a directed graph G where
   - Nodes are samples in the training set;
   - Edges model the relation of ***being-closer*** to the adversarial region of x;
2. For each label ℓ
   - Traverse the graph G
   - Find a ***valid*** path with k samples where ℓ is dominant label;
3. If more than one dominant label is found
   - k-NN is not stable on x;
   - otherwise k-NN is stable on x;

Graph Construction

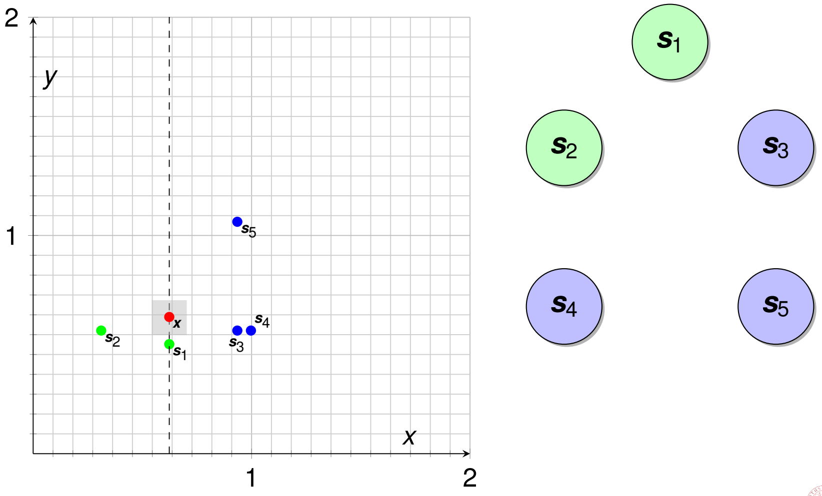


Graph Construction

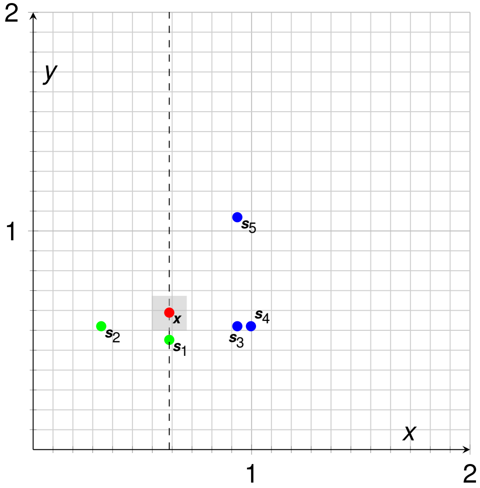


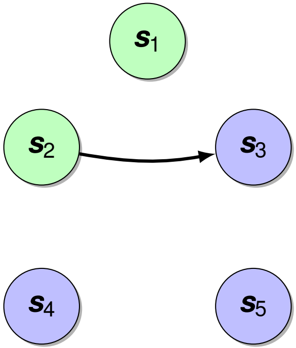


Graph Construction

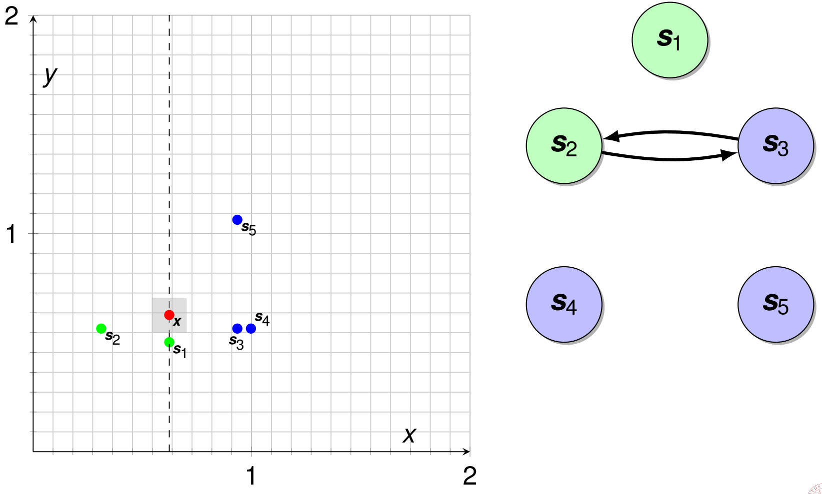


Graph Construction

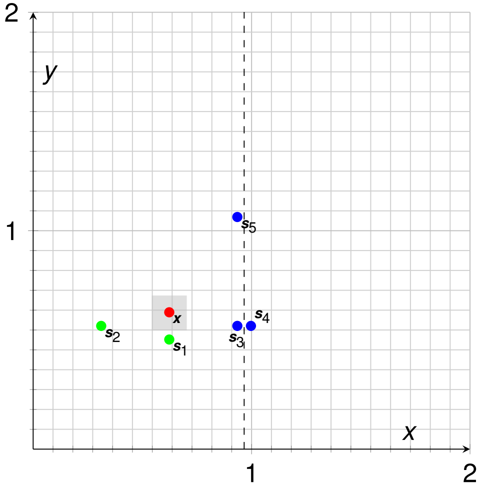


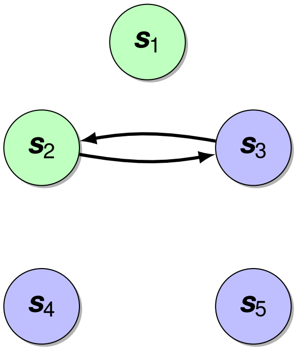


Graph Construction

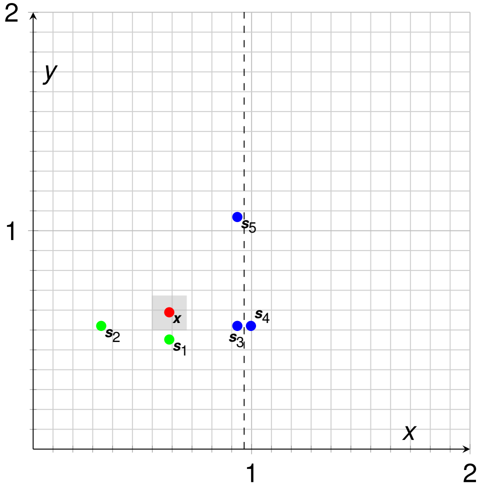


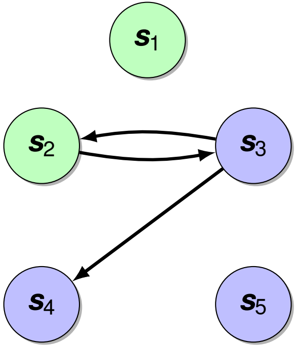
Graph Construction



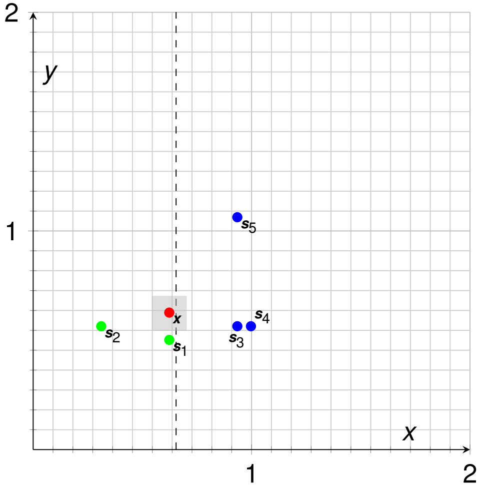


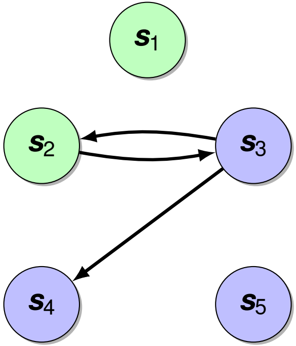
Graph Construction



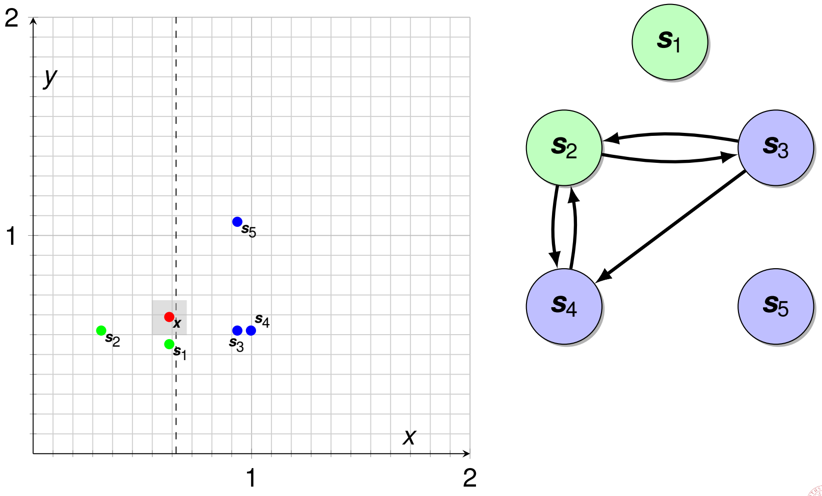


Graph Construction

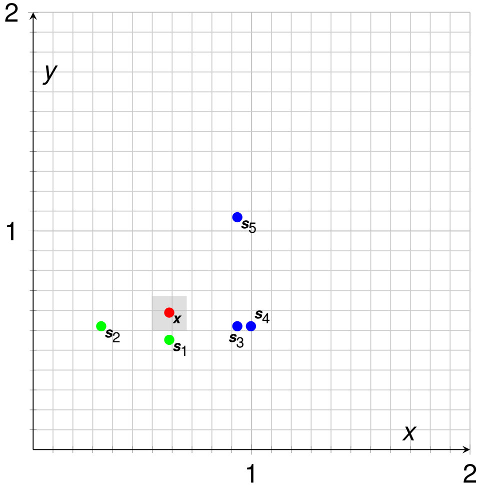


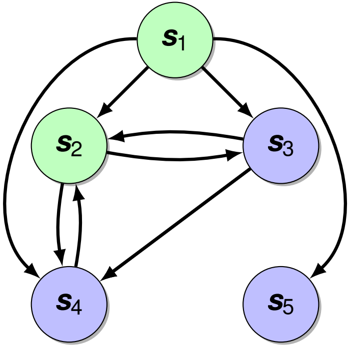


Graph Construction

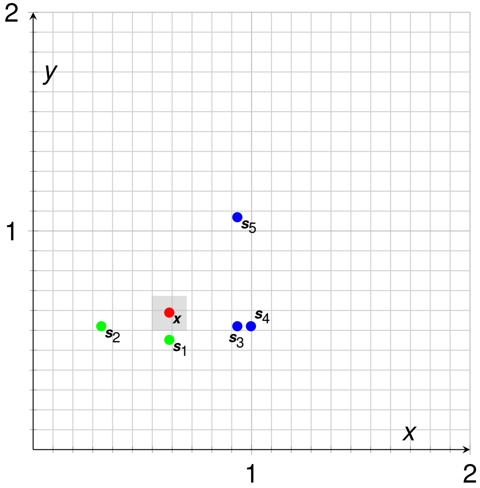


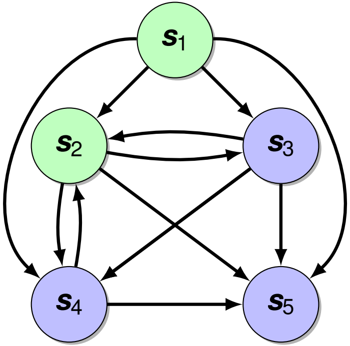
Graph Construction



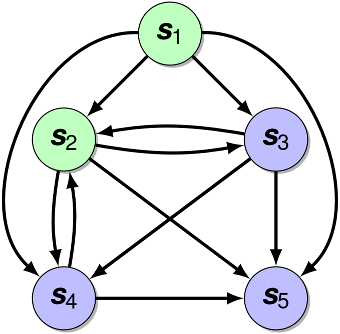


Graph Construction



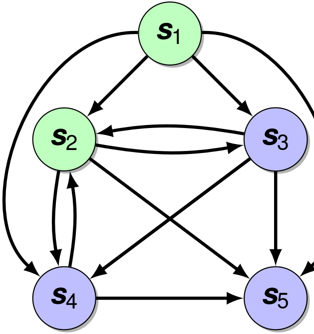


Graph Traversal



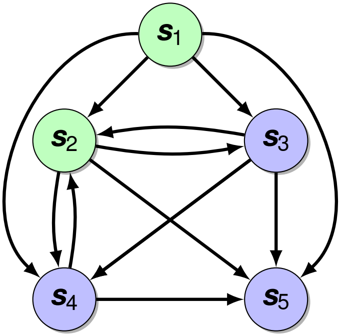
Graph Traversal

$k = 1$



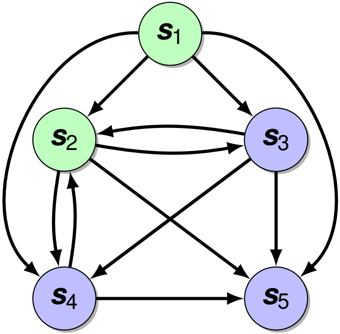
Graph Traversal

$k = 1$ — Labels = $\{green\}$ — Stable = YES



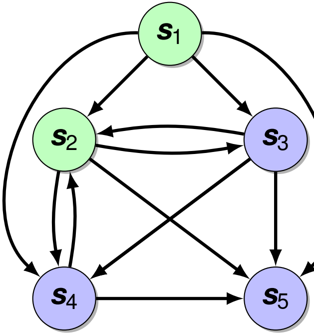
Graph Traversal

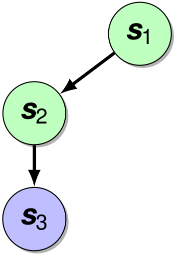
$k = 3$



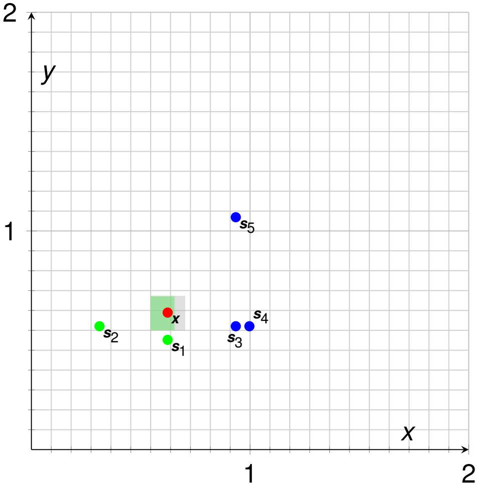
Graph Traversal

$k = 3$



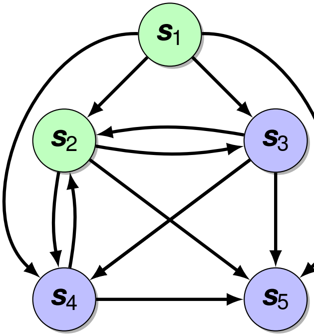


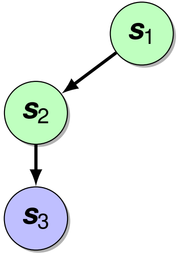
Graph Construction



Graph Traversal

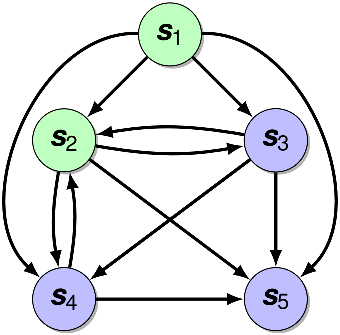
$k = 3$ — Labels = $\{green\}$

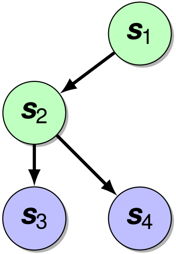




Graph Traversal

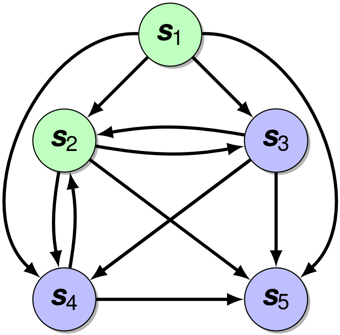
$k = 3$ — Labels = $\{green\}$

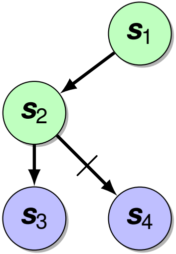




Graph Traversal

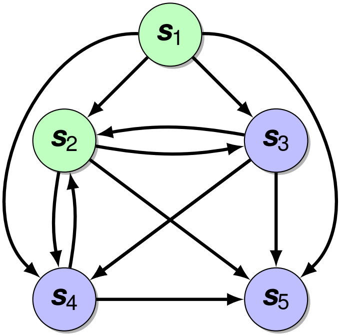
$k = 3$ — Labels = $\{green\}$

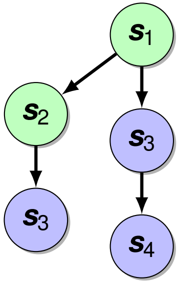




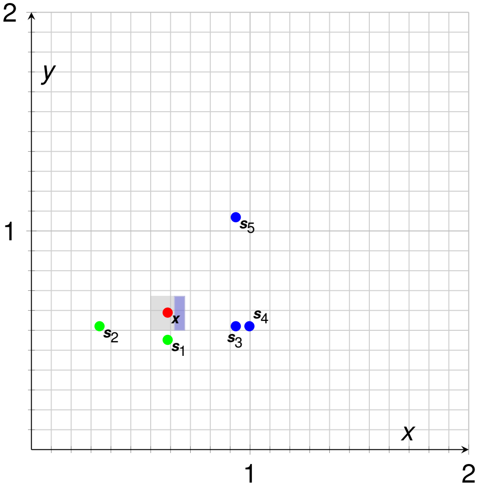
Graph Traversal

$k = 3$ — Labels = $\{green\}$



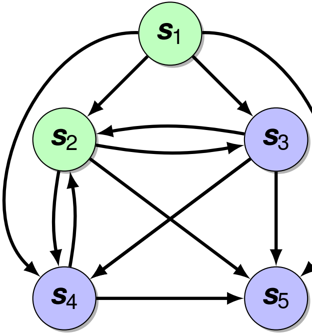


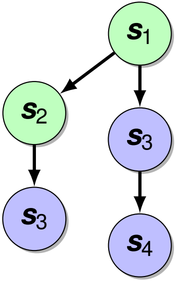
Graph Construction



Graph Traversal

$k = 3$ — Labels = $\{$*green, blue*$\}$ — Stable = NO





Graph Traversal-Summary

- Start traversal of the graph from sample $\boldsymbol{s}_i$ not dominated by any other samples;
- Consider only the paths $[\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_k]$ such that $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_k$ are the closest samples to some $x' \in P^{\epsilon}(\boldsymbol{x})$ than any other samples;

Università degli Studi di Padova

Experimental Evaluation

- Evaluated k-NN robustness using 7 datasets;
- Used $k \in \{1, 3, 5, 7\}$;
- Perturbation region with ϵ up to 0.05;

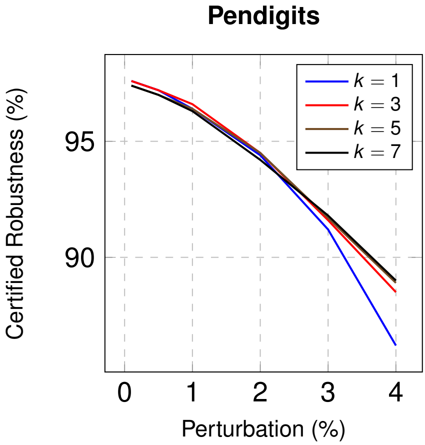
Datasets

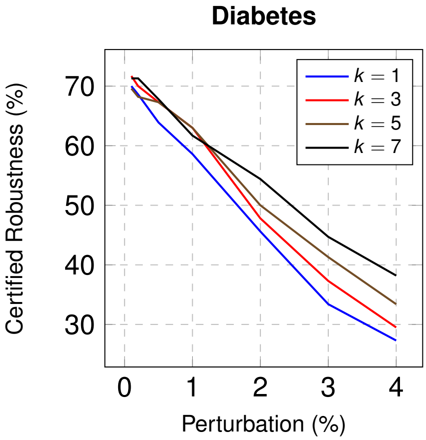
Name	#training	#test	#features	#features (one-hot)	#classes
Australian	483	207	14	39	2
BreastCancer	479	204	10	10	2
Diabetes	556	230	8	8	2
Fourclass	604	258	2	2	2
Letter	15000	5000	16	16	26
Pendigits	7494	3498	16	16	10
Satimage	4435	2000	36	36	6

Preprocessing

- Rows and columns with missing values are dropped;
- When needed, datasets are split into training ($\approx 70 - 80\%$) and test ($\approx 20 - 30\%$) sets;
- Categorical features are one-hot encoded;
- Numerical features are scaled to $[0, 1]$ range.

Robustness Results





Limitations

- Not always able to scale over high value of k or ϵ;
- Much time wasted on finding a path which does not exist;

Name	ϵ	Avg. Time per ϵ (seconds)
Australian	[0.001, 0.05]	2
BreastCancer	[0.001, 0.05]	2.75
Diabetes	[0.001, 0.04]	180
Fourclass	[0.001, 0.04]	1.2
Letter	[0.001, 0.01]	120
Pendigits	[0.001, 0.04]	900
Satimage	[0.001, 0.01]	120

Conclusion

- Developed a novel algorithm to certify the robustness of k-NN;
- Focus on completeness;
- Experimental evaluation showed overall good results;

Future Works

- Leverage high-order voronoi diagram to further reduce certification time;
- Adapt the algorithm to other distance metrics like the Manhattan distance or the more general Minkoswki distance.

Thanks for listening !!

Graph Construction Optimization

Consider only the samples within the hyperball centered in x and radius

$$2\epsilon\sqrt{N} + d$$

where

- ϵ is the radius of the ℓ_∞ ball;
- d the distance between x and its k-th closest sample;
- N is the number of features;

Proposition

Given a perturbation region $P^\epsilon(x)$, the hyperball centered in x with radius $2\epsilon\sqrt{N} + d$ contains the k closest samples of every point $x_0 \in P^\epsilon(x)$