Weight-based Semantic Testing Approach for Neural Networks

Amany Alshareef, Nicolas Berthier, Sven Schewe, Xiaowei Huang

Department of Computer Science
University of Liverpool, UK
Testing goal

- Evidence that the system meets its requirements
- Evidence that the system is error-free

Machine learned software !!

- Data-driven NOT requirements-driven
- Do NOT have a specific control-flow structure
- Most testing techniques propose structural coverage
- Tend at transforming in the input data

Real world high-dimensional data lie on low-dimensional manifolds embedded within the high-dimensional.
→ Not thinking about input domain, instead where the data lie and model that

The research presents a testing approach for neural networks that leverages the learned representations and feature importance to evaluate the test data coverage.

- Analyse the latent features learnt by the model
- Generate additional test cases
Bayesian Network Abstraction Model

A dimensionality reduction technique using feature extraction algorithms to abstract the behaviour of a neural network into a BN.

Constructing a Bayesian Network:

I. Hidden features extraction
   Map from a high-dimensional space into a feature space.

II. Feature space discretization
   Discretise each feature component into finite feature intervals.

III. Probability tables construction
   • Represent the probabilistic distribution of each extracted feature with a node in the BN.
   • Associate each feature with a marginal or a conditional probability table.
BN-based Latent Feature Analysis

Estimate the **importance** of a neural network’s **latent features** by analysing an associated **Bayesian network’s sensitivity** to **distributional shifts**

1. **Probability calculation for input sample under BN**
   Perform the feature projection and discretisation step to input sample to obtain the associated feature intervals, and then calculate their probability belonging to the BN distribution.

2. **Latent features perturbation**
   For each latent feature, randomly shifting its intervals in a selected feature space.

3. **Distance computation**
   Compute the distance between the original probability vector and the probability vector obtained from the perturbed features.
The proposed BN analysis technique to compute the sensitivity of extracted latent features

Sample $X$ $\rightarrow$ Discriminant $\rightarrow$ $Fx$ $\rightarrow$ $Pr(Fx \in B)$

Latent Features Distance

Bayesian Network ($B$)

Compute probabilities of $Fx$ w.r.t the BN

For each latent feature $f(i, j)$ in $Fx$:

Compute probabilities of $F'x_f$ w.r.t the BN

Compute the distance $d_p (Pr(Fx), Pr(F'x_f))$

Higher distribution change $\rightarrow$ Higher importance score

$W_{i,j} = \frac{\delta(P_{ref}, P'_f)}{\sum_{f \in F} \delta(P_{ref}, P'_f)}$

$w_{(1, 1)} = 0.192$

$w_{(2, 1)} = 0.172$
2 Weight-based Testing Metric

Transforming the traditional binary coverage approach to a weighted probability problem and define our coverage metric based on the latent features importance.

- **Weight-based Feature Coverage**

\[
WFCov(\mathcal{B}_N^x) \overset{\text{def}}{=} \sum_{\langle f_{i,j}^k \rangle \in V_{N,x}} w_{\langle f_{i,j}^k \rangle} \cdot \left| \left\{ f_{i,j}^k \in F_{i,j}^\# \mid \mathcal{P}_i \left( f_{i,j}^k \right) \geq \varepsilon \right\} \right|
\]

- **Weight-based Feature Dependence Coverage**

\[
WFdCov(\mathcal{B}_N^x) \overset{\text{def}}{=} \sum_{\langle f_{i,j}^k \rangle \in V_{N,x}} w_{\langle f_{i,j}^k \rangle} \cdot \left| \left\{ f_{i,j}^k, F_{i-1}^\# \in \mathcal{CP}_i \left( f_{i,j}^k, F_{i-1}^\# \right) \geq \varepsilon \right\} \right| \left| \left\{ f_{i,j}^k, F_{i-1}^\# \in \mathcal{P}_i \left( f_{i,j}^k \right) < \varepsilon \right\} \right|
\]

The two feature metrics are combined to produce the generalised weight feature coverage.

\[
WFCovTot(\mathcal{B}_N^x) = \sum_{\langle f_{i,j}^k \rangle \in V_{N,x}} w_{\langle f_{i,j}^k \rangle} \cdot \left\{ \begin{array}{ll}
WFCov(\langle f_{i,j}^k \rangle) & \text{if } i = 1, \\
\frac{1}{2} \left( WFCov(\langle f_{i,j}^k \rangle) + WFdCov(\langle f_{i,j}^k \rangle) \right) & \text{otherwise}.
\end{array} \right.
\]
2 Weight-based Testing Metric

Example 1

\[ \Pr(f_{3,0} < 3) \approx 0.453, \Pr(3 \leq f_{3,0} \leq 5) \approx 0.323, \Pr(f_{3,0} > 5) \approx 0.224. \]

\[ 1 \times 0.173 = 0.173. \]

\[ \text{WFCov}(B_N, x) = 1 \]

Example 2

\[ \varepsilon = 0.01 \]

\[ CP_{f_{3,0}} = \frac{0.1730}{0.8687} \cdot \frac{26}{27} = 0.1917 \]

\[ CP_{f_{3,1}} = \frac{0.2532}{0.8687} \cdot \frac{25}{27} = 0.2875 \]

\[ \text{WFdCov}(B_N, x) = 0.2190 + 0.2532 + 0.1917 + 0.2875 = 0.9514\%. \]
Concolic Test Data Generation Algorithm

Algorithm 1 Test Dataset Generation

Input:
\( \mathcal{N} \) ← DNN under test
\( X \) ← data set
\( \mathcal{B}_{\mathcal{N},X_{\text{train}}} \) ← abstract BN
\( W_f \) ← features sensitivity weights

Output: test inputs \( X_{0} \), coverage

1: \( X_{0} \) ← sampling initial seed test inputs from \( X_{\text{test}} \)
2: \( \mathcal{B}_{\mathcal{N},X_{0}} \) ← initialising the BN prob. tables with \( X_{0} \)
3: \( \text{Tar\_invals} \) ← intervals with prob \( \leq \varepsilon \)
4: for \( i = 1 \) to max iterations do
5: \( t \) ← \( \text{Tar\_invals} \) with highest weight in \( W_f \)
6: select a test input \( s \in X_{0} \)
7: construct an LP problem based on \( t \)
8: solve the optimisation objective:
\[
\min \| (n_{1,1}, \ldots, n_{1,|t_1|}) - (s_{1,1}, \ldots, s_{1,|t_1|}) \|_{\infty}
\]
9: \( s' = (n_{1,1}, \ldots, n_{1,|t_1|}) \)
10: if \( s' \) passes the oracle then
11: \( s' \) ← newly generated test input
12: if \( f_{\mathcal{N}}(s') = f_{\mathcal{N}}(s) \) then
13: \( X_{0} \) ← \( X_{0} \cup \{s'\} \)
14: update \( \mathcal{B}_{\mathcal{N},X_{0}} \) probabilities
15: update coverage
16: else
17: \( s' \) ← adversarial input
18: end if
19: end if
20: end for
Evaluation

1. Datasets and Models
   - The first model targets the Fashion-MNIST classification problem with 89.03% validation accuracy, and the second one targets the CIFAR-10 dataset with 81.00% validation accuracy.
   - The models are reasonably sized, with more than 10 layers, including blocks of convolutional and max-pooling layers, followed by a series of dense layers.
   - The considered layers are the convolutional ReLU, 2d max pooling, and dense ReLU.

2. Experimental Setup
   - Two linear feature extraction techniques were selected: PCA and ICA, with two to five numbers of extracted features for each of the abstracted layers.
   - The Kernel Density Estimation (KDE) and uniform-based discretisation are considered, with varying numbers of the uniform partitions bins that are: one, three, and five.
   - The extended Concolic testing tool is run on both DNNs models with a maximum 100 iterations per run. Each run is initialised with uniformly drawn test sets of 10 and 100 correctly classified inputs.
Results

The overall distribution of initial and the respective final coverage of up to 100 iterations of Concolic test case generation. X-axis indicates the run time in seconds (initial and run time). The vertical lines are the coverage, and the horizontal line on the coverage is the median.
We have introduced a weight-based semantic testing approach that measures how well the DNN is tested by focusing on the important features of the DNN using its abstracted Bayesian network.

- The developed weighted feature metrics achieved higher testing coverage than the original metrics.
- The test generation algorithm is directed to synthesise new input targeting features with higher importance scores.
- Empirically validated the applicability and effectiveness of the proposed weight metrics, which serves as a strong argument in favour of increasing the trustworthy performance of the DNN models.
THANK YOU!