Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

ARTICLE TYPE

Appendix

Islam T. Elgendy | **Robert M. Hierons** | **Phil McMinn**

Computer Science Department, University of Sheffield, Sheffield, UK

Abstract

In this appendix, we provide a more detailed descriptions of the similarity metrics used in our study.

1 | INTRODUCTION

This appendix contains detailed information about all the similarity metrics reported in our study.

2 | RQ2: SIMILARITY METRICS

What similarity metrics have been used in the literature? Which ones have been used the most, and why?

All studies that apply diversity in their approaches use some metric to measure the level of similarity or diversity. The similarity can be calculated for inputs, outputs, or any other testing artefacts.

There are many similarity metrics used in the literature, and we found 79 metrics in the collected papers. Some of these metrics are well-known, like Euclidean distance, Hamming distance, and so on, while others are more specific to certain types of subject domains or new metrics. We categorized the similarity metrics into two groups. The first group consists of the generic similarity metrics that originated from other fields. The second group are specialised metrics in Software Engineering to measure similarity based on information acquired from software programs, or metrics proposed to solve a Software Engineering problem.

Table 1 presents the generic similarity metrics, while Table 2 lists the specialised similarity metrics in software engineering. For both tables, records are ordered by usage popularity in the literature and then by alphabetical order. For each similarity metric, we provide a citation for more details, a short description, how many papers used that metric, and all papers using the metric in our collection. The citation after the specialised similarity metric in Table 2 is the paper that introduced the specialised metric.

The three most popular similarity metrics are Euclidean distance, Edit distance, and Jaccard distance used in 35, 31, and 30 papers, respectively. Numeric programs are used by many researchers to evaluate their techniques and Euclidean distance is a natural choice for such programs. Also, with string data, the Edit distance is very popular to use. Furthermore, Jaccard distance is widely used when the testing artefacts can be represented as sets, with an example being software product lines in which a product can be seen as being a set of features.

TABLE 1: A list of the generic similarity metrics citing the source, a brief description, the number of papers using them and citations

ID	Metric & Source	Description	Total	Papers

G1	Euclidean distance[1]	The square root of the sum of the squared differences between the vectors X and Y. The formula is: $Euc(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$	35	2,3,4,5,6,7,8 9,10,11,12,13,14,15 16,17,18,19,20,21,22 23,24,25,26,27,28,29 30,31,32,33,34,35,36
G2	Edit distance[37]	The minimum number of edits *(insertions, deletions or substitutions)* required to change one string into the other. It takes into consideration that parts of the strings can be similar even if not in corresponding places, and can work with strings of different sizes.	31	38,39,40,41,42,43,10,11 44,45,13,46,47,16,48 49,50,51,52,21,53,54,55 56,26,32,57,58,59,60
G3	Jaccard distance[61]	The ratio of intersection over union between two sets A and B of values. The formula is: $Jac(A,B) = 1 - \frac{\lvert A \cap B\rvert}{\lvert A \cup B\rvert} = 1 - \frac{\lvert A \cap B\rvert}{\lvert A\rvert + \lvert B\rvert - \lvert A \cap B\rvert}$ Sometimes expressed[34] as: $Jac(A,B) = 1 - \frac{A.B}{A.B + \omega(\lVert A \rVert^2 + \lVert B \rVert^2 - 2(A.B))}$ with with $\omega = 1$.	31	62,43,44,10,11,45,63,64,51 65,16,66,49,67,68,69 52,70,71,72,54,73,74 75,76,77,34,78,79,80,81
G4	Hamming distance[82]	The number of times when the corresponding characters in two strings are different. Huang et al.[83] refer to this as "Overlap" distance.	19	84,85,86,10,11,13,51 87,49,83,21,70,55 56,32,88,89,90,91
G5	Manhattan distance[92]	The sum of the absolute differences between two vectors X and Y. The formula is: $Man(X,Y) = \sum_{i=1}^{n} \lvert x_i - y_i\rvert$	18	42,85,93,94,13,16 66,18,21,95,71,72 22,28,31,32,96,97
G6	Normalised compression distance[98]	An approximation of the Kolmogorov complexity using real-world compressors.	15	99,100,101,45,102,103,13 104,105,106,16,66,52,27,31
G7	Cosine similarity[107]	The cosine of the angle of two vectors X and Y. The formula is: $Cosine(X,Y) = \frac{\sum_{i=1}^{n} x_i \times y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \times \sqrt{\sum_{i=1}^{n} y_i^2}}$	13	108,109,10,11,101,52,110 22,111,32,112,77,34
G8	Tree edit distance[113]	The minimum number of edit operations required to change one tree into the other.	4	73,114,115,116
G9	Locality-sensitive hashing[117]	A technique that maps similar strings or inputs to the same hash code with high probability to get a fast estimation of the dissimilarity between two subjects.	3	66,75,32
G10	Geometric diversity[118]	The measurement of feature similarity between two feature vectors given an input sample.	3	99,100,119
G11	Gower-Legendre distance[120]	A variant of the Jaccard Index (G3) where the weight ω is 1/2.	3	51,55,34
G12	Needleman-Wunsch distance[121]	Originally used in bioinformatics to align protein or nucleotide sequences, and can be used to identify similarities between two test cases by encoding them.	3	49,67,51

G13	Crowding distance[122]	A measure of how far a chromosome or an individual is from the rest of the population.	2	[123,124]
G14	Isolated subTree distance[125]	A variation of tree edit distance (G8), where disjoint subtrees are mapped to similar disjoint subtrees of another set.	2	[114,115]
G15	Jaro-Winkler distance[126]	A variation of the Jaro distance (G26) that adds more weight in strings starting with the exact match characters.	2	[84,44]
G16	L2-test[127]	The distance between a uniform distribution and a sampled distribution by checking if the sampled distribution is ϵ-far from uniformity.	2	[128,129]
G17	Mahalanobis distance[130]	The distance between a point and a distribution.	2	[28,31]
G18	Smith-Waterman distance[131]	The alignment of local sequences for determining similar regions between two strings of nucleic acid sequences or protein sequences.	2	[49,51]
G19	Sokal-Sneath distance[132]	A variant of the Jaccard Index (G3) where the weight ω is 2.	2	[51,34]
G20	Wasserstein distance[133]	The difference between two frequency distributions over a region, which is also known as the earth mover's distance.	2	[134,135]
G21	Anti-Dice[136]	It is a variation of the Jaccard distance, that measure the similarity between two sets where the denominator is the sum of the lengths of the two sets rather than the union.	1	[137]
G22	Bilingual Evaluation Understudy[138]	A numerical translation closeness metric of a machine translation to a human translation.	1	[139]
G23	Canberra distance[140]	A weighted version of the Manhattan distance (G5), in which each term in the sum is normalised.	1	[28]
G24	Chebyshev distance[141]	The greatest difference between two points in two vectors along any coordinate dimension.	1	[28]
G25	Fractional distance[142]	A variation of the Euclidean distance (G1) to deal with multi-dimensional space.	1	[143]
G26	Hellinger distance[144]	The difference between two distributions with Hellinger integral[145].	1	[134]
G27	Hill-numbers[146]	A measure originally used in ecology that considers both species richness and species abundances in a sample.	1	[147]
G28	Jaro distance[148]	The number of matching characters and the number of transpositions (i.e. matching characters but not in order) between two strings.	1	[44]
G29	Jeffrey divergence[149]	A derived distribution from the Kullback-Leibler Divergence (G30) that is symmetric and more robust to noises.	1	[28]
G30	Jensen-Shannon distance[150]	An improved version of Kullback-Leibler Divergence (G30) to measure the similarity of two probability distributions. The metric is symmetric and always has a finite value.	1	[134]
G31	Kronecker delta[151]	A discrete function of two variables that is one if they are equal, 0 otherwise.	1	[152]

G32	Kullback-Leibler divergence[153]	The expected value of the logarithmic difference between two probability distributions, but it is not symmetric.	1	[134]
G33	Mean-square-error[154]	The average squared difference between the values predicted from a model and the actual values.	1	[155]
G34	Modified trigonometric distance[156]	A modified version of the trigonometric distance (G46) with a greater degree of accuracy for points of larger magnitude of values.	1	[28]
G35	N-gram models[157]	A contiguous sequence of *n* items from a given sample of text or speech.	1	[158]
G36	Ochiai coefficient[159]	An approximation of program semantics using passing and failing test cases.	1	[139]
G37	Proportional distance[160]	The sum of squares of the difference between two vectors over the difference between the maximum and minimum values.	1	[34]
G38	Shannon's Diversity Index[161]	A measure of diversity used in Ecology to measure the variety and abundance of species in a defined unit of study.	1	[27]
G39	Singular value decomposition[162]	An estimate of where the evolution is going in search-based approaches, by monitoring the movements of individuals across different generations.	1	[163]
G40	Statistic value X^2[164]	A distance function that emphasizes large absolute difference existing between the feature values.	1	[28]
G41	String-Kernels[165]	The inner product between two strings by counting the occurrences of common substrings in the two strings.	1	[34]
G42	Sellers algorithm[166]	A variation of the edit distance (G2) to find a substring in another string with at most *k* edit operations.	1	[44]
G43	Tree Bottom-Up[167]	A similarity measure for tree structured data based on the bottom-up maximum common subtree isomorphism algorithm.	1	[116]
G44	Tree Kernels[168]	A similarity measure for tree structured data by summing the contribution of fragments (e.g. whole subtree, or subsets of a tree) to the overall similarity by the tree.	1	[169]
G45	Tree Top-Down[167]	A similarity measure for tree structured data based on the top-down maximum common subtree isomorphism algorithm.	1	[116]
G46	Trigonometric distance[156]	A normalised distance between two points used in image matching. The distance between two vectors X and Y is $\sum_{i=1}^{n} \sin(\arctan \lvert x_i - y_i \rvert)$	1	[28]
G47	Word mover's distance[170]	The minimum amount of distance that the embedded words of one document need to be moved to reach the embedded words of another document.	1	[77]

TABLE 2: A list of the specialised Software Engineering similarity metrics.

ID	Metric & Source	Description	Total	Papers

S1	Identical transition distance[171]	The number of identical transitions between two finite state machines divided by the average length of paths.	6	[171,172,48,49,50,51]
S2	Test set diameter (TSDm)[105]	An extension of the pairwise normalised compression distance (G6) to multisets.	6	[105,66,17,114,115,101]
S3	Identical state distance[48]	The number of identical states between two paths of finite state machines divided by their average number of states.	3	[48,50,51]
S4	Trigger-based distance[48]	An extension of identical transition similarity (S1) to account for triggers in the transitions.	3	[48,50,51]
S5	Average population diameter[173]	The average distance between all vectors in a population, where the distance between two vectors is the difference of their lengths.	2	[173,174]
S6	Approach level[175]	The number of mismatched branch predicates to reach the target branch.	2	[176,177]
S7	Basic counting[56]	The overlapping occurrences of method calls between two failing sequences of method calls extracted from execution traces of tests.	2	[51,56]
S8	Distinguishing mutation adequacy[178]	An assessment of the diversity of mutants' behaviour based on the mutants' killing information.	2	[178,179]
S9	Extended subTree distance[114]	A variation of isolated subtree distance (G13) with different mapping conditions.	2	[114,115]
S10	Path distance[180]	The size of the intersection between the two paths of multisets of trees.	2	[114,115]
S11	[GUI] State similarity[181]	The difference between the values of two GUI states using the widgets of the GUI.	2	[181,182]
S12	Test diversity[183]	A hybrid measure calculating the difference between two test cases in terms of branches covered, variation of the data inputs, and standard deviation between conditions covered.	2	[183,184]
S13	[Graph model diversity] Symmetric distance[111]	The difference between two models in a domain-specific language, where it is calculated as the number of "shapes" contained exclusively in one of the models but not both.	1	[111]
S14	Text uniqueness[185]	Text matching between two strings, where a string is unique if no other string matches it.	1	[185]
S15	Tree Combined[116]	A similarity measure for tree structured data that combines bottom-up (G43) and top-down (G45) common subtree isomorphism algorithms.	1	[116]
S16	Achieved coverage of pools[186]	The number of items selected from a pool of values for a program's variables over the time spent using that pool of values.	1	[186]
S17	Accuracy-based performance measure[187]	The proportion of correctly predicted test inputs to all the test inputs for a DNN.	1	[187]
S18	[Test behavioural similarity] Accuracy (acc)[188]	The percentage of tests that fail or pass together, calculated as the number of correct predictions divided by the total number of predictions in a confusion matrix.	1	[102]

S19	Average cyclomatic complexity per method (ACCM)[189]	Cyclomatic complexity is the number of independent paths in a program or method. ACCM is calculated by computing the number of independent paths within each method and then taking the sum, over all methods, of these values.	1	[190]
S20	Code complexity (cm)[3]	Consists of three types of information (Lines of code, Nested Block Depth, and Cyclomatic Complexity) derived from the source code to measure similarity.	1	[3]
S21	[GUI similarity] $CONTeSSi(n)$[109]	The differences of the frequencies between the past n executed events of one test suite to another test suite.	1	[109]
S22	Distance entropy[191]	The distribution of tests in a set represented in a graph using the minimum weight set (i.e. the set of vertices or edges in a weighted graph that collectively has the smallest sum of weights).	1	[191]
S23	Diversification distance[192]	A measure in Output Diversified Sampling strategy[192] that measures the distance between iteration output and the original output.	1	[193]
S24	Enhanced Jaro-Winkler[84]	A hybrid metric between Jaro-Winkler (G14) and Hamming Distance (G4) that considers the deselected features from Hamming distance combined into the Jaro distance equation.	1	[84]
S25	Graph edit distance[194]	The minimum number of edit operations required to make two graphs identical.	1	[76]
S26	Matthew's correlation coefficient (MCC)[195]	A more accurate measure of tests behavioural similarity than accuracy (S17) that accounts for both true positives and true negatives.	1	[102]
S27	Probabilistic type tree[196]	A tree structure to represent a probability distribution over the types.	1	[196]
S28	Response for class (RFC)[189]	The sum of the number of methods inside the class and the number of external methods used by the class.	1	[190]
S29	Syntax-tree similarity[197]	The structural similarity between two sentences represented as "syntax trees" by comparing the tree topologies, node positions, and the types of grammatical relationships.	1	[197]
S30	Traces[132]	The difference between two execution paths, that takes into account the branches covered and the number of times these branches were covered.	1	[198]
S31	Weighted distance function[199]	The number of statements covered by one test case but not the other, and the difference of the execution times between the two test cases.	1	[199]
S32	Extensible access control markup language (XACML) Similarity[200]	The distance between the requests attributes' values of two XACML test cases and the difference between their policies.	1	[200]

REFERENCES

1. Alpha W. EuclideanDistance - Wolfram: Alpha. https://mathworld.wolfram.com/EuclideanMetric.html; Retrived Sep. 19, 2024.
2. Albunian N, Fraser G, Sudholt D. Measuring and maintaining population diversity in search-based unit test generation. In: *Proceedings of the International Symposium on Search Based Software Engineering (SSBSE)* 2020:153–168. doi: 10.1007/978-3-030-59762-7_11.

3. Arafeen MJ, Do H. Test case prioritization using requirements-based clustering. In: *Proceedings of the International Conference on Software Testing, Validation and Verification (ICST)* 2013:312–321. doi: 10.1109/ICST.2013.12.
4. Arrieta A, Wang S, Markiegi U, Arruabarrena A, Etxeberria L, Sagardui G. Pareto efficient multi-objective black-box test case selection for simulation-based testing. *Information and Software Technology.* 2019;114:137–154. doi: 10.1016/j.infsof.2019.06.009.
5. Attaoui M, Fahmy H, Pastore F, Briand L. Black-box safety analysis and retraining of DNNs based on feature extraction and clustering. *ACM Transactions on Software Engineering and Methodology.* 2023;32(3):1–40. doi: 10.1145/3550271.
6. Bueno PM, Wong WE, Jino M. Automatic test data generation using particle systems. In: *Proceedings of the Symposium on Applied Computing (SAC)* 2008:809–814. doi: 10.1145/1363686.1363871.
7. Bueno PM, Jino M, Wong WE. Diversity oriented test data generation using metaheuristic search techniques. *Information sciences.* 2014;259:490–509. doi: 10.1016/j.ins.2011.01.025.
8. Carlson R, Do H, Denton A. A clustering approach to improving test case prioritization: An industrial case study. In: *Proceedings of the International Conference on Software Maintenance (ICSM)* 2011:382–391. doi: 10.1109/ICSM.2011.6080805.
9. Chetouane N, Wotawa F, Felbinger H, Nica M. On using k-means clustering for test suite reduction. In: *Proceedings of the Workshop on Testing: Academia-Industry Collaboration, Practice and Research Techniques (TAIC)* 2020:380–385. doi: 10.1109/ICSTW50294.2020.00068.
10. Coviello C, Romano S, Scanniello G. An empirical study of inadequate and adequate test suite reduction approaches. In: *Proceedings of the International Symposium on Empirical Software Engineering and Measurement (ESEM)* 2018:1–10. doi: 10.1145/3239235.3240497.
11. Coviello C, Romano S, Scanniello G, Marchetto A, Antoniol G, Corazza A. Clustering support for inadequate test suite reduction. In: *Proceedings of the International Conference on Software Analysis, Evolution and Reengineering (SANER)* 2018:95–105. doi: 10.1109/SANER.2018.8330200.
12. Cruciani E, Miranda B, Verdecchia R, Bertolino A. Scalable approaches for test suite reduction. In: *Proceedings of the International Conference on Software Engineering (ICSE)* 2019:419–429. doi: 10.1109/ICSE.2019.00055.
13. Elgendy I, Hierons R, McMinn P. Evaluating string distance metrics for reducing automatically generated test suites. In: *Proceedings of the International Conference on Automation of Software Test (AST)* 2024:171–181.
14. Farzat Fd. Test case selection method for emergency changes. In: *Proceedings of the International Symposium on Search Based Software Engineering (SSBSE)* 2010:31–35. doi: 10.1109/SSBSE.2010.13.
15. Greca R, Miranda B, Gligoric M, Bertolino A. Comparing and combining file-based selection and similarity-based prioritization towards regression test orchestration. In: *Proceedings of the International Conference on Automation of Software Test (AST)* 2022:115–125. doi: 10.1145/3524481.3527223.
16. Guarnieri GF, Pizzoleto AV, Ferrari FC. An automated framework for cost reduction of mutation testing based on program similarity. In: *Proceedings of the International Workshop on Mutation Analysis (Mutation)* 2022:179–188. doi: 10.1109/ICSTW55395.2022.00041.
17. Haghighatkhah A, Mäntylä M, Oivo M, Kuvaja P. Test prioritization in continuous integration environments. *Journal of Systems and Software.* 2018;146:80–98. doi: 10.1016/j.jss.2018.08.061.
18. Hemmati H, Fang Z, Mantyla MV. Prioritizing manual test cases in traditional and rapid release environments. In: *Proceedings of the International Conference on Software Testing, Verification and Validation (ICST)* 2015:1–10. doi: 10.1109/ICST.2015.7102602.
19. Kim J, Feldt R, Yoo S. Guiding deep learning system testing using surprise adequacy. In: *Proceedings of the International Conference on Software Engineering (ICSE)* 2019:1039–1049. doi: 10.1109/ICSE.2019.00108.
20. Lan W, Cui Z, Zhang J, Yang J, Gu X. Fuzz testing based on seed diversity analysis. In: *Proceedings of the International Conference on Systems, Man, and Cybernetics (SMC)* 2023:145–150. doi: 10.1109/SMC53992.2023.10394486.
21. Ledru Y, Petrenko A, Boroday S, Mandran N. Prioritizing test cases with string distances. *Automated Software Engineering.* 2012;19(1):65–95. doi: 10.1007/s10515-011-0093-0.
22. Mahdieh M, Mirian-Hosseinabadi S, Mahdieh M. Test case prioritization using test case diversification and fault-proneness estimations. *Automated Software Engineering.* 2022;29(2):50. doi: 10.1007/s10515-022-00344-y.
23. Matinnejad R, Nejati S, Briand LC, Bruckmann T. Effective test suites for mixed discrete-continuous stateflow controllers. In: *Proceedings of the Joint Meeting on Foundations of Software Engineering (ESEC/FSE)* 2015:84–95. doi: 10.1145/2786805.2786818.
24. Matinnejad R, Nejati S, Briand LC, Bruckmann T. Automated test suite generation for time-continuous Simulink models. In: *Proceedings of the International Conference on Software Engineering (ICSE)* 2016:595–606. doi: 10.1145/2884781.2884797.
25. Matinnejad R, Nejati S, Briand LC, Bruckmann T. Test generation and test prioritization for Simulink models with dynamic behavior. *IEEE Transactions on Software Engineering.* 2019;45(9):919–944. doi: 10.1109/TSE.2018.2811489.
26. Nass M, Alégroth E, Feldt R, Leotta M, Ricca F. Similarity-based web element localization for robust test automation. *ACM Transactions on Software Engineering and Methodology.* 2023;32(3):1–30. doi: 10.1145/3571855.
27. Neelofar N, Aleti A. Towards reliable AI: Adequacy metrics for ensuring the quality of system-level testing of autonomous vehicles. In: *Proceedings of the International Conference on Software Engineering (ICSE)* 2024:1–12. doi: 10.1145/3597503.3623314.
28. Nunes FL, Delamaro ME, Gonçalves VM, Lauretto MDS. CBIR based testing oracles: An experimental evaluation of similarity functions. *International Journal of Software Engineering and Knowledge Engineering.* 2015;25(08):1271–1306. doi: 10.1142/S0218194015500254.
29. Pei Y, Christi A, Fern X, Groce A, Wong W. Taming a fuzzer using delta debugging trails. In: *Proceedings of the International Conference on Data Mining Workshop (ICDMW)* 2014:840–843. doi: 10.1109/ICDMW.2014.58.
30. Riccio V, Humbatova N, Jahangirova G, Tonella P. Deepmetis: Augmenting a deep learning test set to increase its mutation score. In: *Proceedings of the International Conference on Automated Software Engineering (ASE)* 2021:355–367. doi: 10.1109/ASE51524.2021.9678764.
31. Rogstad E, Briand L, Torkar R. Test case selection for black-box regression testing of database applications. *Information and Software Technology.* 2013;55(10):1781–1795. doi: 10.1016/j.infsof.2013.04.004.
32. Shahbazi A, Miller J. Black-box string test case generation through a multi-objective optimization. *IEEE Transactions on Software Engineering.* 2015;42(4):361–378. doi: 10.1109/TSE.2015.2487958.
33. Shimari K, Tanaka M, Ishio T, Matsushita M, Inoue K, Takanezawa S. Selecting test cases based on similarity of runtime information: A Case study of an industrial simulator. In: *Proceedings of the International Conference on Software Maintenance and Evolution (ICSME)* 2022:564–567. doi: 10.1109/ICSME55016.2022.00077.
34. Wang R, Jiang S, Chen D. Similarity-based regression test case prioritization. In: *Proceedings of the International Conference on Software Engineering and Knowledge Engineering (SEKE)* 2015:358–363. doi: 10.18293/seke2015-115.

35. Yu X, Jia K, Hu W, Tian J, Xiang J. Black-box test case prioritization using log analysis and test case diversity. In: *Proceedings of the International Symposium on Software Reliability Engineering Workshops (ISSREW)* 2023:186–191. doi: 10.1109/ISSREW60843.2023.00072.
36. Zhao X, Wang Z, Fan X, Wang Z. A clustering-bayesian network based approach for test case prioritization. In: *Proceedings of the Annual Computer Software and Applications Conference (COMPSAC)*. 3. 2015:542–547. doi: 10.1109/COMPSAC.2015.154.
37. Levenshtein VI. Binary codes capable of correcting deletions, insertions, and reversals. 1966;10(8):707–710.
38. Adigun JG, H. TP, Camilli M, Felderer M. Risk-driven online testing and test case diversity analysis for ML-enabled critical systems. In: *Proceedings of the International Symposium on Software Reliability Engineering (ISSRE)* 2023:344–354. doi: 10.1109/ISSRE59848.2023.00017.
39. Almulla H, Gay G. Generating diverse test suites for Gson through adaptive fitness function selection. In: *Proceedings of the International Symposium on Search Based Software Engineering (SSBSE)* 2020:246–252. doi: 10.1007/978-3-030-59762-7_18.
40. Asoudeh N, Labiche Y. A multi-objective genetic algorithm for generating test suites from extended finite state machines. In: *Proceedings of the International Symposium on Search Based Software Engineering (SSBSE)* 2013:288–293. doi: 10.1007/978-3-642-39742-4_26.
41. Biagiola M, Stocco A, Ricca F, Tonella P. Diversity-based web test generation. In: *Proceedings of the Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)* 2019:142–153. doi: 10.1145/3338906.3338970.
42. Boussaa M, Barais O, Sunyé G, Baudry B. A novelty search approach for automatic test data generation. In: *Proceedings of the International Workshop on Search-Based Software Testing (SBST)* 2015:40–43. doi: 10.1109/SBST.2015.17.
43. Cao J, Li M, Li Y, Wen M, Cheung S, Chen H. SemMT: A semantic-based testing approach for machine translation systems. *ACM Transactions on Software Engineering and Methodology.* 2022;31(2):1–36. doi: 10.1145/3490488.
44. Coutinho AVB, Cartaxo EG, Lima Machado dPD. Analysis of distance functions for similarity-based test suite reduction in the context of model-based testing. *Software Quality Journal.* 2016;24(2):407–445. doi: 10.1007/s11219-014-9265-z.
45. Oliveira Neto dFG, Ahmad A, Leifler O, Sandahl K, Enoiu E. Improving continuous integration with similarity-based test case selection. In: *Proceedings of the International Workshop on Automation of Software Test (AST)* 2018:39–45. doi: 10.1145/3194733.3194744.
46. Fang C, Chen Z, Wu K, Zhao Z. Similarity-based test case prioritization using ordered sequences of program entities. *Software Quality Journal.* 2014;22(2):335–361. doi: 10.1007/s11219-013-9224-0.
47. Flemström D, Afzal W, Sundmark D. Exploring test overlap in system integration: An industrial case study. In: *Proceedings of the Euromicro Conference on Software Engineering and Advanced Applications (SEAA)* 2016:303–312. doi: 10.1109/SEAA.2016.34.
48. Hemmati H, Briand L, Arcuri A, Ali S. An enhanced test case selection approach for model-based testing: An industrial case study. In: *Proceedings of the International Symposium on Foundations of Software Engineering (FSE)* 2010:267–276. doi: 10.1145/1882291.1882331.
49. Hemmati H, Briand L. An industrial investigation of similarity measures for model-based test case selection. In: *Proceedings of the International Symposium on Software Reliability Engineering (ISSRE)* 2010:141–150. doi: 10.1109/ISSRE.2010.9.
50. Hemmati H, Arcuri A, Briand L. Reducing the cost of model-based testing through test case diversity. In: *Proceedings of the International Conference on Testing Software and Systems (ICTSS)* 2010:63–78. doi: 10.1007/978-3-642-16573-3_6.
51. Hemmati H, Arcuri A, Briand L. Achieving scalable model-based testing through test case diversity. *ACM Transactions on Software Engineering and Methodology.* 2013;22(1):1–42. doi: 10.1145/2430536.2430540.
52. Khojah R, Chao CH, Oliveira Neto dFG. Evaluating the trade-offs of text-based diversity in test prioritisation. In: *Proceedings of the International Conference on Automation of Software Test (AST)* 2023:168-178. doi: 10.1109/AST58925.2023.00021.
53. Ma L, Wu P, Chen TY. Diversity driven adaptive test generation for concurrent data structures. *Information and software technology.* 2018;103:162–173. doi: 10.1016/j.infsof.2018.07.001.
54. Mariani L, Mohebbi A, Pezzè M, Terragni V. Semantic matching of GUI events for test reuse: Are we there yet?. In: *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)* 2021:177–190. doi: 10.1145/3460319.3464827.
55. Mondal D, Hemmati H, Durocher S. Exploring test suite diversification and code coverage in multi-objective test case selection. In: *Proceedings of the International Conference on Software Testing, Verification and Validation (ICST)* 2015:1–10. doi: 10.1109/ICST.2015.7102588.
56. Noor TB, Hemmati H. A similarity-based approach for test case prioritization using historical failure data. In: *Proceedings of the International Symposium on Software Reliability Engineering (ISSRE)* 2015:58–68. doi: 10.1109/ISSRE.2015.7381799.
57. Shimmi S, Rahimi M. Leveraging code-test co-evolution patterns for automated test case recommendation. In: *Proceedings of the International Conference on Automation of Software Test (AST)* 2022:65–76. doi: 10.1145/3524481.3527222.
58. Wang W, Wu S, Li Z, Zhao R. Parallel evolutionary test case generation for web applications. *Information and Software Technology.* 2023;155:107113. doi: 10.1016/j.infsof.2022.107113.
59. Wang W, Guo J, Li B, Shang Y, Zhao R. EFSM model-based testing for android applications. *International Journal of Software Engineering and Knowledge Engineering.* 2024;34(04):597–621. doi: 10.1142/S0218194023500638.
60. Zhang D, Liu D, Lei Y, et al. SimFuzz: Test case similarity directed deep fuzzing. *Journal of Systems and Software.* 2012;85(1):102–111. doi: 10.1016/j.jss.2011.07.028.
61. Jaccard P. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin de la Société Vaudoise des Sciences Naturelles.* 1901;37:547–579.
62. Aman H, Nakano T, Ogasawara H, Kawahara M. A test case recommendation method based on morphological analysis, clustering and the Mahalanobis-Taguchi method. In: *Proceedings of the Workshop on Testing: Academia-Industry Collaboration, Practice and Research Techniques (TAICPART)* 2017:29–35. doi: 10.1109/ICSTW.2017.9.
63. Oliveira Neto dFG, Feldt R, Erlenhov L, Nunes JBDS. Visualizing test diversity to support test optimisation. In: *Proceedings of the Asia-Pacific Software Engineering Conference (APSEC)* 2018:149–158. doi: 10.1109/APSEC.2018.00029.
64. Fischer S, Lopez-Herrejon RE, Ramler R, Egyed A. A preliminary empirical assessment of similarity for combinatorial iteraction testing of software product lines. In: *Proceedings of the International Workshop on Search-Based Software Testing (SBST)* 2016:15–18. doi: 10.1145/2897010.289701.
65. Gao X, Feng Y, Yin Y, Liu Z, Chen Z, Xu B. Adaptive test selection for deep neural networks. In: *Proceedings of the International Conference on Software Engineering (ICSE)* 2022:73–85. doi: 10.1145/3510003.3510232.
66. Haghighatkhah A, Mäntylä M, Oivo M, Kuvaja P. Test case prioritization using test similarities. In: *Proceedings of the International Conference on Product-Focused Software Process Improvement (PROFES)* 2018:243–259. doi: 10.1007/978-3-030-03673-7_18.
67. Hemmati H, Arcuri A, Briand L. Empirical investigation of the effects of test suite properties on similarity-based test case selection. In: *Proceedings of the International Conference on Software Testing, Verification and Validation (ICST)* 2011:327–336. doi: 10.1109/ICST.2011.12.

68. Henard C, Papadakis M, Perrouin G, Klein J, Le Traon Y. Assessing software product line testing via model-based mutation: An application to similarity testing. In: *Proceedings of the Advances in Model Based Testing Workshop (A-MOST)* 2013:188–197. doi: 10.1109/ICSTW.2013.30.
69. Henard C, Papadakis M, Perrouin G, Klein J, Heymans P, Le Traon Y. Bypassing the combinatorial explosion: Using similarity to generate and prioritize t-wise test configurations for software product lines. *IEEE Transactions on Software Engineering.* 2014;40(7):650–670. doi: 10.1109/TSE.2014.2327020.
70. Lee M, Cha S, Oh H. Learning seed-adaptive mutation strategies for greybox fuzzing. In: *Proceedings of the International Conference on Software Engineering (ICSE)* 2023:384–396. doi: 10.1109/ICSE48619.2023.00043.
71. Liu B, Lucia , Nejati S, Briand LC. Improving fault localization for Simulink models using search-based testing and prediction models. In: *International Conference on Software Analysis, Evolution and Reengineering (SANER)* 2017:359–370. doi: 10.1109/SANER.2017.7884636.
72. Liu B, Nejati S, Lucia , Briand LC. Effective fault localization of automotive Simulink models: achieving the trade-off between test oracle effort and fault localization accuracy. *Empirical Software Engineering.* 2019;24(1):444–490. doi: 10.1007/s10664-018-9611-z.
73. Mei L, Cai Y, Jia C, Jiang B, Chan W. Test pair selection for test case prioritization in regression testing for WS-BPEL programs. *International Journal of Web Services Research (IJWSR).* 2013;10(1):73–102. doi: 10.4018/jwsr.2013010104.
74. Miranda B, Bertolino A. Social coverage for customized test adequacy and selection criteria. In: *Proceedings of the International Workshop on Automation of Software Test (AST)* 2014:22–28. doi: 10.1145/2593501.2593505.
75. Miranda B, Cruciani E, Verdecchia R, Bertolino A. FAST approaches to scalable similarity-based test case prioritization. In: *Proceedings of the International Conference on Software Engineering (ICSE)* 2018:222–232. doi: 10.1145/3180155.3180210.
76. Tang Y, Jiang H, Zhou Z, Li X, Ren Z, Kong W. Detecting compiler warning defects via diversity-guided program mutation. *IEEE Transactions on Software Engineering.* 2022;48(11):4411-4432. doi: 10.1109/TSE.2021.3119186.
77. Viggiato M, Paas D, Buzon C, Bezemer C. Identifying similar test cases that are specified in natural language. *IEEE Transactions on Software Engineering.* 2023;49(3):1027-1043. doi: 10.1109/TSE.2022.3170272.
78. Xiang Y, Huang H, Li M, Li S, Yang X. Looking for novelty in search-based software product line testing. *IEEE Transactions on Software Engineering.* 2021;1(1):1–1. doi: 10.1109/TSE.2021.3057853.
79. You YS, Huang CY, Peng KL, Hsu CJ. Evaluation and analysis of spectrum-based fault localization with modified similarity coefficients for software debugging. In: *Proceedings of the Annual Computer Software and Applications Conference (COMPSAC)* 2013:180–189. doi: 10.1109/COMPSAC.2013.32.
80. Zhang Z, Xie X. On the investigation of essential diversities for deep learning testing criteria. In: *Proceedings of the International Conference on Software Quality, Reliability and Security (QRS)* 2019:394–405. doi: 10.1109/QRS.2019.00056.
81. Zhao L, Zhang Z, Wang L, Yin X. A fault localization framework to alleviate the impact of execution similarity. *International Journal of Software Engineering and Knowledge Engineering.* 2013;23(07):963–998. doi: 10.1142/S0218194013500289.
82. Hamming RW. Error detecting and error correcting codes. *The Bell System Technical Journal.* 1950;29(2):147–160. doi: 10.1002/j.1538-7305.1950.tb00463.x.
83. Huang R, Zhou Y, Zong W, Towey D, Chen J. An empirical comparison of similarity measures for abstract test case prioritization. In: *Proceedings of the Annual Computer Software and Applications Conference (COMPSAC).* 1. 2017:3–12. doi: 10.1109/COMPSAC.2017.271.
84. Abd Halim S, Jawawi DNA, Sahak M. Similarity distance measure and prioritization algorithm for test case prioritization in software product line testing. *Journal of Information and Communication Technology.* 2019;18(1):57–75.
85. Cao Y, Zhou ZQ, Chen TY. On the correlation between the effectiveness of metamorphic relations and dissimilarities of test case executions. In: *Proceedings of the International Conference on Quality Software (QSIC)* 2013:153–162. doi: 10.1109/QSIC.2013.43.
86. Cheng M, Zhou Y, Xie X. Behavexplor: Behavior diversity guided testing for autonomous driving systems. In: *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)* 2023:488–500. doi: 10.1145/3597926.3598072.
87. Feng Y, Chen Z, Jones JA, Fang C, Xu B. Test report prioritization to assist crowdsourced testing. In: *Proceedings of the Joint Meeting on Foundations of Software Engineering* 2015:225–236. doi: 10.1145/2786805.2786862.
88. Wang H, Chan W. Weaving context sensitivity into test suite construction. In: *Proceedings of the International Conference on Automated Software Engineering (ASE)* 2009:610–614. doi: 10.1109/ASE.2009.79.
89. Wang H, Chan W, Tse T. Improving the effectiveness of testing pervasive software via context diversity. *Transactions on Autonomous and Adaptive Systems.* 2014;9(2):1–28. doi: 10.1145/2620000.
90. Xie X, Xu B, Shi L, Nie C. Genetic test case generation for path-oriented testing. *Journal of Software.* 2009;20(12):3117–3136. doi: 10.3724/SP.J.1001.2009.00580.
91. Yoo S, Harman M, Tonella P, Susi A. Clustering test cases to achieve effective and scalable prioritisation incorporating expert knowledge. In: *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)* 2009:201–212. doi: 10.1145/1572272.1572296.
92. Alpha W. ManhattanDistance - Wolfram: Alpha. https://www.wolframalpha.com/input?i=ManhattanDistance; Retrived Sep. 19, 2024.
93. Chen J, Wang G, Hao D, Xiong Y, Zhang H, Zhang L. History-guided configuration diversification for compiler test-program generation. In: *Proceedings of the International Conference on Automated Software Engineering (ASE)* 2019:305–316. doi: 10.1109/ASE.2019.00037.
94. Chen J, Gu Y, Cai S, Chen H, Chen J. KS-TCP: An efficient test case prioritization approach based on k-medoids and similarity. In: *Proceedings of the International Symposium on Software Reliability Engineering Workshops (ISSREW)* 2021:105–110. doi: 10.1109/ISSREW53611.2021.00051.
95. Li X, Guo S, Cheng H, Jiang H. Simulink compiler testing via configuration diversification with reinforcement learning. *IEEE Transactions on Reliability.* 2024;73(2):1060-1074. doi: 10.1109/TR.2023.3317643.
96. Zohdinasab T, Riccio V, Gambi A, Tonella P. Deephyperion: Exploring the feature space of deep learning-based systems through illumination search. In: *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)* 2021:79–90. doi: 10.1145/3460319.3464811.
97. Zohdinasab T, Riccio V, Gambi A, Tonella P. Efficient and effective feature space exploration for testing deep learning systems. *ACM Transactions on Software Engineering and Methodology.* 2023;32(2):1–38. doi: 10.1145/3544792.
98. Cilibrasi R, Vitányi PM. Clustering by compression. *IEEE Transactions on Information theory.* 2005;51(4):1523–1545. doi: 10.1109/TIT.2005.844059.
99. Aghababaeyan Z, Abdellatif M, Briand L, Ramesh S, Bagherzadeh M. Black-box testing of deep neural networks through test case diversity. *IEEE Transactions on Software Engineering.* 2023;49(5):3182-3204. doi: 10.1109/TSE.2023.3243522.
100. Aghababaeyan Z, Abdellatif M, Dadkhah M, Briand L. Deepgd: A multi-objective black-box test selection approach for deep neural networks. *ACM Transactions on Software Engineering and Methodology.* 2024;33(6). doi: 10.1145/3644388.

101. Dai H, Sun C, Liu H, Zhang X. DFuzzer: diversity-driven seed queue construction of fuzzing for deep learning models. *IEEE Transactions on Reliability.* 2024;73(2):1075-1089. doi: 10.1109/TR.2023.3322406.
102. Oliveira Neto dFG, Dobslaw F, Feldt R. Using mutation testing to measure behavioural test diversity. In: *Proceedings of the International Workshop on Mutation Analysis (Mutation)* 2020:254–263. doi: 10.1109/ICSTW50294.2020.00051.
103. Dobslaw F, Oliveira N. dFG, Feldt R. Boundary value exploration for software analysis. In: *Proceedings of the Workshop on NEXt Level of Test Automation (NEXTA)* 2020:346–353. doi: 10.1109/ICSTW50294.2020.00062.
104. Feldt R, Torkar R, Gorschek T, Afzal W. Searching for cognitively diverse tests: Towards universal test diversity metrics. In: *Proceedings of the International Conference on Software Testing Verification and Validation Workshop (ICSTW)* 2008:178–186. doi: 10.1109/ICSTW.2008.36.
105. Feldt R, Poulding S, Clark D, Yoo S. Test set diameter: Quantifying the diversity of sets of test cases. In: *Proceedings of the International Conference on Software Testing, Verification and Validation (ICST)* 2016:223–233. doi: 10.1109/ICST.2016.33.
106. Feldt R, Dobslaw F. Towards automated boundary value testing with program derivatives and search. In: *Proceedings of the International Symposium on Search Based Software Engineering (SSBSE)* 2019:155–163. doi: 10.1007/978-3-030-27455-9_11.
107. Alpha W. CosineDistance - Wolfram: Alpha. https://www.wolframalpha.com/input?i=CosineDistance; Retrived Sep. 19, 2024.
108. Azizi M. A tag-based recommender system for regression test case prioritization. In: *Proceedings of the International Workshop on Software Test Architecture (InSTA)* 2021:146–157. doi: 10.1109/ICSTW52544.2021.00035.
109. Brooks PA, Memon AM. Introducing a test suite similarity metric for event sequence-based test cases. In: *Proceedings of the International Conference on Software Maintenance (ICSM)* 2009:243–252. doi: 10.1109/ICSM.2009.5306305.
110. Lin J, Wang F, Chu P. Using semantic similarity in crawling-based web application testing. In: *Proceedings of the International Conference on Software Testing, Verification and Validation (ICST)* 2017:138–148. doi: 10.1109/ICST.2017.20.
111. Semeráth O, Farkas R, Bergmann G, Varró D. Diversity of graph models and graph generators in mutation testing. *International Journal on Software Tools for Technology Transfer.* 2020;22(1):57–78. doi: 10.1007/s10009-019-00530-6.
112. Sondhi D, Jobanputra M, Rani D, Purandare S, Sharma S, Purandare R. Mining similar methods for test adaptation. *IEEE Transactions on Software Engineering.* 2022;48(07):2262-2276. doi: 10.1109/TSE.2021.3057163.
113. Zhang K, Shasha D. Simple fast algorithms for the editing distance between trees and related problems. *SIAM journal on computing.* 1989;18(6):1245–1262. doi: 10.1137/0218082.
114. Shahbazi A, Miller J. Extended subtree: A new similarity function for tree structured data. *Transactions on Knowledge and Data Engineering.* 2014;26(4):864 - 877. doi: 10.1109/TKDE.2013.53.
115. Shahbazi A, Panahandeh M, Miller J. Black-box tree test case generation through diversity. *Automated Software Engineering.* 2018;25(3):531–568. doi: 10.1007/s10515-018-0232-y.
116. Pan R, Ghaleb TA, Briand L. Atm: Black-box test case minimization based on test code similarity and evolutionary search. In: *Proceedings of the International Conference on Software Engineering (ICSE)* 2023:1700–1711. doi: 10.1109/ICSE48619.2023.00146.
117. Shakhnarovich G, Darrell T, Indyk P. *Nearest-neighbor methods in learning and vision: theory and practice (neural information processing).* The MIT press, 2006.
118. Kulesza A, Taskar B. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning.* 2012;5(2–3):123–286. doi: 10.1561/2200000044.
119. Jiang Z, Li H, Wang R. Efficient generation of valid test inputs for deep neural networks via gradient search. *Journal of Software: Evolution and Process.* 2023;n/a(n/a):e2550. doi: 10.1002/smr.2550.
120. Gower JC, Legendre P. Metric and Euclidean properties of dissimilarity coefficients. *Journal of classification.* 1986;3:5–48. doi: 10.1007/BF01896809.
121. Needleman SB, Wunsch CD. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology.* 1970;48(3):443–453. doi: 10.1016/0022-2836(70)90057-4.
122. Mahfoud SW. *Niching methods for genetic algorithms.* University of Illinois at Urbana-Champaign, 1995.
123. Panichella A, Oliveto R, Di Penta M, De Lucia A. Improving multi-objective test case selection by injecting diversity in genetic algorithms. *IEEE Transactions on Software Engineering.* 2015;41(4):358–383. doi: 10.1109/TSE.2014.2364175.
124. Scalabrino S, Grano G, Nucci DD, Oliveto R, Lucia AD. Search-based testing of procedural programs: Iterative single-target or multi-target approach?. In: *Proceedings of the International Symposium on Search Based Software Engineering (SSBSE)* 2016:64–79. doi: 10.1007/978-3-319-47106-8_5.
125. Tanaka E, Tanaka K. The tree-to-tree editing problem. *International Journal of pattern recognition and artificial intelligence.* 1988;2(02):221–240. doi: 10.1016/0020-0190(77)90064-3.
126. Winkler WE. The state of record linkage and current research problems. In: *Statistical Research Division, US Census Bureau* 1999:1–15.
127. Goldreich O. *On testing expansion in bounded-degree graphs.* 20, 2000.
128. Menéndez HD, Jahangirova G, Sarro F, Tonella P, Clark D. Diversifying focused testing for unit testing. *ACM Transactions on Software Engineering and Methodology.* 2021;30(4):1–24. doi: 10.1145/3447265.
129. Menendez HD, Clark D. Hashing fuzzing: Introducing input diversity to improve crash detection. *IEEE Transactions on Software Engineering.* 2022;48(09):3540-3553. doi: 10.1109/TSE.2021.3100858.
130. Chandra MP, others . On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India.* 1936;2(1):49–55.
131. Smith TF, Waterman MS. Identification of common molecular subsequences. *Journal of molecular biology.* 1981;147(1):195–197. doi: 10.1016/0022-2836(81)90087-5.
132. Zalewski M. American fuzzy lop. 2014.
133. Arjovsky M, Chintala S, Bottou L. Wasserstein generative adversarial networks. In: *Proceedings of the International Conference on Machine Learning (ICML)* 2017:214–223.
134. Xie X, Yin P, Chen S. Boosting the revealing of detected violations in deep learning testing: A diversity-guided method. In: *Proceedings of the International Conference on Automated Software Engineering (ASE)* 2022:1–13. doi: 10.1145/3551349.3556919.
135. Wang W, Chen Z, Zheng Z, Wang H. An adaptive fuzzing method based on transformer and protocol similarity mutation. *Computers & Security.* 2023;129(C):103197. doi: 10.1016/j.cose.2023.103197.
136. Gusfield D. Algorithms on stings, trees, and sequences: Computer science and computational biology. *Acm Sigact News.* 1997;28(4):41–60. doi: 10.1145/270563.571472.

137. Xiang Y, Huang H, Li S, Li M, Luo C, Yang X. Automated test suite generation for software product lines based on quality-diversity optimization. *ACM Transactions on Software Engineering and Methodology.* 2023;33(2):1–52. doi: 10.1145/3628158.
138. Papineni K, Roukos S, Ward T, Zhu WJ. BLEU: a method for automatic evaluation of machine translation. In: *Proceedings of the annual meeting of the Association for Computational Linguistics* 2002:311–318. doi: 10.3115/1073083.1073135.
139. Ojdanic M, Garg A, Khanfir A, Degiovanni R, Papadakis M, Le Traon Y. Syntactic versus semantic similarity of artificial and real faults in mutation testing studies. *IEEE Transactions on Software Engineering.* 2023;49(7):3922–3938. doi: 10.1109/TSE.2023.3277564.
140. Alpha W. CanberraDistance - Wolfram: Alpha. https://reference.wolfram.com/language/ref/CanberraDistance.html; Retrived Sep. 19, 2024.
141. Alpha W. ChebyshevDistance - Wolfram: Alpha. https://reference.wolfram.com/language/ref/ChebyshevDistance.html; Retrived Sep. 19, 2024.
142. Aggarwal CC, Hinneburg A, Keim DA. On the surprising behavior of distance metrics in high dimensional space. In: *Database Theory—ICDT* 2001:420–434. doi: 10.1007/3-540-44503-X_27.
143. Marculescu B, Feldt R, Torkar R. Using exploration focused techniques to augment search-based software testing: An experimental evaluation. In: *Proceedings of the International Conference on Software Testing, Verification and Validation (ICST)* 2016:69–79. doi: 10.1109/ICST.2016.26.
144. mathematics oE. Hellinger distance - Encyclopedia of mathematics. https://encyclopediaofmath.org/index.php?title=Hellinger_distance; Retrived Sep. 19, 2024.
145. Hellinger E. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen.. *Journal für die reine und angewandte Mathematik.* 1909;1909(136):210–271. doi: 10.1515/crll.1909.136.210.
146. Hill MO. Diversity and evenness: a unifying notation and its consequences. *Ecology.* 1973;54(2):427–432. doi: 10.2307/1934352.
147. Nguyen HL, Grunske L. BEDIVFUZZ: Integrating behavioral diversity into generator-based fuzzing. In: *Proceedings of the International Conference on Software Engineering (ICSE)* 2022:249-261. doi: 10.1145/3510003.3510182.
148. Jaro MA. Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida. *Journal of the American Statistical Association.* 1989;84(406):414–420. doi: 10.1080/01621459.1989.10478785.
149. Jeffreys H. *The theory of probability*. OuP Oxford, 1998.
150. Menéndez M, Pardo J, Pardo L, Pardo M. The jensen-shannon divergence. *Journal of the Franklin Institute.* 1997;334(2):307–318. doi: 10.1016/S0016-0032(96)00063-4.
151. Alpha W. Kronecker Delta - Wolfram: Alpha. https://mathworld.wolfram.com/KroneckerDelta.html; Retrived Sep. 19, 2024.
152. Kichigin DY. A method of test-suite reduction for regression integration testing. *Programming and Computer Software.* 2009;35(5):282–290. doi: 10.1134/S0361768809050041.
153. Kullback S, Leibler R. On information and sufficiencyannals of mathematical statistics. *MathSciNet MATH.* 1951;22(n/a):79–86.
154. Britannica F. Mean squared error - Britannica. https://www.britannica.com/science/mean-squared-error; Retrived Sep. 19, 2024.
155. Mosin V, Staron M, Durisic D, Oliveira Neto dFG, Pandey SK, Koppisetty AC. Comparing input prioritization techniques for testing deep learning algorithms. In: *Proceedings of the Euromicro Conference on Software Engineering and Advanced Applications (SEAA)* 2022:76–83. doi: 10.1109/SEAA56994.2022.00020.
156. Li Z, Hou K, Li H. Similarity measurement based on trigonometric function distance. In: *International Symposium on Pervasive Computing and Applications* 2006:227–231. doi: 10.1109/SPCA.2006.297573.
157. Broder AZ, Glassman SC, Manasse MS, Zweig G. Syntactic clustering of the web. *Computer networks and ISDN systems.* 1997;29(8-13):1157–1166. doi: 10.1016/S0169-7552(97)00031-7.
158. Leveau J, Blanc X, Réveillére L, Falleri J, Rouvoy R. Fostering the diversity of exploratory testing in web applications. In: *Proceedings of the International Conference on Software Testing, Validation and Verification (ICST)* 2020:164-174. doi: 10.1109/ICST46399.2020.00026.
159. Ochiai A. Zoogeographic studies on the soleoid fishes found in Japan and its neighbouring regions. *Bulletin of Japanese Society of Scientific Fisheries.* 1957;22:526–530.
160. Dickinson W, Leon D, Fodgurski A. Finding failures by cluster analysis of execution profiles. In: *Proceedings of the International Conference on Software Engineering (ICSE)* 2001:339–348. doi: 10.1109/ICSE.2001.919107.
161. Magurran AE. Measuring biological diversity. *Current Biology.* 2021;31(19):R1174–R1177. doi: 10.1016/j.cub.2021.07.049.
162. Stewart GW. On the early history of the singular value decomposition. *SIAM review.* 1993;35(4):551–566. doi: 10.1137/1035134.
163. Kifetew FM, Panichella A, De Lucia A, Oliveto R, Tonella P. Orthogonal exploration of the search space in evolutionary test case generation. In: *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)* 2013:257–267. doi: 10.1145/2483760.2483789.
164. Bugatti PH, Traina AJ, Traina Jr C. Assessing the best integration between distance-function and image-feature to answer similarity queries. In: *Proceedings of the Symposium on Applied Computing (SAC)* 2008:1225–1230. doi: 10.1145/1363686.1363969.
165. Leslie C, Eskin E, Weston J, Noble WS. Mismatch string kernels for SVM protein classification. *Advances in neural information processing systems.* 2004;5(n/a):1441–1448.
166. Sellers PH. The theory and computation of evolutionary distances: pattern recognition. *Journal of algorithms.* 1980;1(4):359–373. doi: 10.1016/0196-6774(80)90016-4.
167. Valiente G. *Algorithms on trees and graphs*. 112. Springer, 2002.
168. Collins M, Duffy N. Convolution kernels for natural language. *Advances in neural information processing systems.* 2001;14. doi: 10.5555/2980539.2980621.
169. Altiero F, Corazza A, Di Martino S, Peron A, Libero Lucio Starace L. Regression test prioritization leveraging source code similarity with tree kernels. *Journal of Software: Evolution and Process.* 2024:e2653. doi: 10.1002/smr.2653.
170. Kusner M, Sun Y, Kolkin N, Weinberger K. From word embeddings to document distances. In: *Proceedings of the International Conference on Machine Learning (ICML)* 2015:957–966.
171. Cartaxo EG, Machado PD, Oliveira Neto dFG. On the use of a similarity function for test case selection in the context of model-based testing. *Software Testing, Verification and Reliability.* 2009;21(2):75–100. doi: 10.1002/stvr.413.
172. Oliveira Neto dFG, Torkar R, Machado PD. Full modification coverage through automatic similarity-based test case selection. *Information and Software Technology.* 2016;80:124–137. doi: 10.1016/j.infsof.2016.08.008.
173. Vogel T, Tran C, Grunske L. Does diversity improve the test suite generation for mobile applications?. In: *Proceedings of the International Symposium on Search Based Software Engineering (SSBSE)* 2019:58–74. doi: 10.1007/978-3-030-27455-9_5.
174. Vogel T, Tran C, Grunske L. A comprehensive empirical evaluation of generating test suites for mobile applications with diversity. *Information and Software Technology.* 2021;130:106436. doi: 10.1016/j.infsof.2020.106436.

175. McMinn P. Search-based software test data generation: a survey. *Software testing, Verification and reliability.* 2004;14(2):105–156. doi: 10.1002/stvr.294.
176. Alshraideh M, Mahafzah BA, Al-Sharaeh S. A multiple-population genetic algorithm for branch coverage test data generation. *Software Quality Journal.* 2011;19:489–513. doi: 10.1007/s11219-010-9117-4.
177. Cai G, Su Q, Hu Z. Binary searching iterative algorithm for generating test cases to cover paths. *Applied Soft Computing.* 2021;113:107910. doi: 10.1016/j.asoc.2021.107910.
178. Shin D, Yoo S, Bae D. Diversity-aware mutation adequacy criterion for improving fault detection capability. In: *Proceedings of the International Workshop on Mutation Analysis (Mutation)* 2016:122–131. doi: 10.1109/ICSTW.2016.37.
179. Shin D, Yoo S, Bae D. A theoretical and empirical study of diversity-aware mutation adequacy criterion. *IEEE Transactions on Software Engineering.* 2018;44(10):914–931. doi: 10.1109/TSE.2017.2732347.
180. Buttler D. A short survey of document structure similarity algorithms. tech. rep., Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States); 2004.
181. Feng J, Yin BB, Cai KY, Yu ZX. 3-way GUI test cases generation based on event-wise partitioning. In: *Proceedings of the International Conference on Quality Software (QSIC)* 2012:89–97. doi: 10.1109/QSIC.2012.42.
182. He Z, Bai C. GUI test case prioritization by state-coverage criterion. In: *Proceedings of the International Workshop on Automation of Software Test (AST)* 2015:18–22. doi: 10.1109/AST.2015.11.
183. Nikolik B. Test diversity. *Information and Software Technology.* 2006;48(11):1083–1094. doi: 10.1016/j.infsof.2006.02.001.
184. Marchetto A, Tonella P. Search-based testing of Ajax web applications. In: *Proceedings of the International Symposium on Search Based Software Engineering (SSBSE)* 2009:3–12. doi: 10.1109/SSBSE.2009.13.
185. Alshahwan N, Harman M. Coverage and fault detection of the output-uniqueness test selection criteria. In: *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)* 2014:181–192. doi: 10.1145/2610384.2610413.
186. Yatoh K, Sakamoto K, Ishikawa F, Honiden S. Feedback-controlled random test generation. In: *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)* 2015:316–326. doi: 10.1145/2771783.2771805.
187. Zhao C, Mu Y, Chen X, Zhao J, Ju X, Wang G. Can test input selection methods for deep neural network guarantee test diversity? A large-scale empirical study. *Information and Software Technology.* 2022;150:106982. doi: 10.1016/j.infsof.2022.106982.
188. Metz CE. Basic principles of ROC analysis. *Seminars in nuclear medicine.* 1978;8(4):283–298. doi: 10.1016/S0001-2998(78)80014-2.
189. Chidamber SR, Kemerer CF. A metrics suite for object oriented design. *IEEE Transactions on software engineering.* 1994;20(6):476–493. doi: 10.1109/32.295895.
190. Pizzoleto AV, Ferrari FC, Dallilo LD, Offutt J. SiMut: Exploring program similarity to support the cost reduction of mutation testing. In: *Proceedings of the International Workshop on Mutation Analysis (Mutation)* 2020:264–273. doi: 10.1109/ICSTW50294.2020.00052.
191. Shi Q, Chen Z, Fang C, Feng Y, Xu B. Measuring the diversity of a test set with distance entropy. *Transactions on Reliability.* 2015;65(1):19–27. doi: 10.1109/TR.2015.2434953.
192. Tashiro Y, Song Y, Ermon S. Diversity can be transferred: Output diversification for white-and black-box attacks. *Advances in neural information processing systems.* 2020;33:4536–4548.
193. Kong W. Transcend adversarial examples: Diversified adversarial attacks to test deep learning model. In: *Proceedings of the International Conference on Computer Design (ICCD)* 2023:13–20. doi: 10.1109/ICCD58817.2023.00013.
194. Sanfeliu A, Fu KS. A distance measure between attributed relational graphs for pattern recognition. *IEEE transactions on systems, man, and cybernetics.* 1983;SMC-13(3):353–362. doi: 10.1109/TSMC.1983.6313167.
195. Matthews BW. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure.* 1975;405(2):442–451. doi: 10.1016/0005-2795(75)90109-9.
196. Poulding S, Feldt R. Automated random testing in multiple dispatch languages. In: *Proceedings of the International Conference on Software Testing, Verification and Validation (ICST)* 2017:333–344. doi: 10.1109/ICST.2017.37.
197. Masuda S, Matsuodani T, Tsuda K. Syntax-tree similarity for test-case derivability in software requirements. In: *Proceedings of the International Workshop on Software Test Architecture (InSTA)* 2021:162–172. doi: 10.1109/ICSTW52544.2021.00037.
198. Reddy S, Lemieux C, Padhye R, Sen K. Quickly generating diverse valid test inputs with reinforcement learning. In: *Proceedings of the International Conference on Software Engineering (ICSE)* 2020:1410–1421. doi: 10.1145/3377811.3380399.
199. Beena R, Sarala S. Multi objective test case minimization collaborated with clustering and minimal hitting set. *Journal of Theoretical and Applied Information Technology.* 2014;69(1):200–210.
200. Bertolino A, Daoudagh S, El Kateb D, et al. Similarity testing for access control. *Information and Software Technology.* 2015;58:355–372. doi: 10.1016/j.infsof.2014.07.003.