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

A more detailed similarity metrics and explanation of the search operationalization.

1 | INTRODUCTION

This appendix contains detailed information about the exact search strings used in every search engine and 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 70 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. Figure **??** shows the distribution of similarity metrics used in DBT papers.

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. As shown in Figure **??**, 80.1% of the DBT papers used generic similarity metrics, while only 19.9% used specialised metrics. The largest portion is the "other generic metrics", which makes up 20.8%, but it contains 32 generic metrics. These metrics are coded G8 to G40 in Table 1. The second-largest portion in the pie chart is the "other specialised metrics" which contains 28 specialised metrics coded S3 to S30 in Table 2.

The three most popular similarity metrics are Euclidean distance, Jaccard distance, and Edit distance used in 27, 26, and 26 papers, respectively. Numeric programs are used by many researchers to evaluate their techniques and Euclidean distance is a natural choice for such programs. Also, 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. Furthermore, if inputs are strings then one might use Edit distance.

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: $Euc(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$	27	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
G2	Jaccard distance [29]	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, (e.g.[27]) as: $Jac(A,B) = 1 - \frac{A.B}{A.B + \omega(\parallel A \parallel^2 + \parallel B \parallel^2 - 2(A.B))}$ with with $\omega = 1$.	26	30,31,9,10,32,33,14 34,35,36,37,38,39,40,41 42,43,44,45,46,47,27 48,49,50,51
G3	Edit distance [52]	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.	26	53,54,55,56,30,31,9 10,32,57,58,14,59,35 60,61,17,62,41,63,64 25,65,66,67,68
G4	Hamming distance [69]	The number of times when the corresponding characters in two strings are different. Some (e.g.[70]) refer to this as "Overlap" distance.	15	71,72,9,10,73,35,70,17 63,64,25,74,75,76,77
G5	Manhattan distance [78]	The sum of the absolute differences between two vectors X and Y. The formula is: $Man(X,Y) = \sum_{i=1}^{n} \lvert x_i - y_i\rvert$	14	56,72,79,80,14,34,16 17,39,40,18,22,24,25
G6	Cosine similarity [81]	The cosine of the angle of two vectors X and Y. The formula is: $Cosine(X,Y) = \frac{\sum_{i=1}^{n} x_i \times y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \times \sqrt{\sum_{i=1}^{n} y_i^2}}$	11	82,83,9,10,84,18 85,25,86,47,27
G7	Normalised compression distance [87]	An approximation of the Kolmogorov complexity using real-world compressors.	9	88,32,89,90,91 92,14,34,24
G8	Tree edit distance [93]	The minimum number of edit operations required to change one tree into the other.	3	42,94,95
G9	Locality-sensitive hashing [96]	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	34,44,25
G10	Crowding distance [97]	A measure of how far a chromosome or an individual is from the rest of the population.	2	98,99
G11	Geometric diversity [100]	The measurement of feature similarity between two feature vectors given an input sample.	2	88,101
G12	Gower-Legendre distance [102]	A variant of the Jaccard Index (G2) where the weight ω is 1/2.	2	63,27

G13	Isolated subTree distance [103]	A variation of tree edit distance (G8), where disjoint subtrees are mapped to similar disjoint subtrees of another set.	2	94,95
G14	Jaro-Winkler distance [104]	A variation of the Jaro distance (G23) that adds more weight in strings starting with the exact match characters.	2	71,31
G15	L2-test [105]	The distance between a uniform distribution and a sampled distribution by checking if the sampled distribution is ϵ-far from uniformity.	2	106,107
G16	Mahalanobis distance [108]	The distance between a point and a distribution.	2	22,24
G17	Needleman-Wunsch distance [109]	Originally used in bioinformatics to align protein or nucleotide sequences, and can be used to identify similarities between two test cases by encoding them.	2	35,36
G18	Canberra distance [110]	A weighted version of the Manhattan distance (G5), in which each term in the sum is normalised.	1	22
G19	Chebyshev distance [111]	The greatest difference between two points in two vectors along any coordinate dimension.	1	22
G20	Fractional distance [112]	A variation of the Euclidean distance (G1) to deal with multi-dimensional space.	1	113
G21	Hellinger distance [114]	The difference between two distributions with Hellinger integral [115].	1	116
G22	Hill-numbers [117]	A measure originally used in ecology that considers both species richness and species abundances in a sample.	1	118
G23	Jaro distance [119]	The number of matching characters and the number of transpositions (i.e. matching characters but not in order) between two strings.	1	31
G24	Jeffrey divergence [120]	A derived distribution from the Kullback-Leibler Divergence (G27) that is symmetric and more robust to noises.	1	22
G25	Jensen-Shannon distance [121]	An improved version of Kullback-Leibler Divergence (G27) to measure the similarity of two probability distributions. The metric is symmetric and always has a finite value.	1	116
G26	Kronecker delta [122]	A discrete function of two variables that is one if they are equal, 0 otherwise.	1	123
G27	Kullback-Leibler divergence [124]	The expected value of the logarithmic difference between two probability distributions, but it is not symmetric.	1	116
G28	Mean-square-error [125]	The average squared difference between the values predicted from a model and the actual values.	1	126
G29	Modified trigonometric distance [127]	A modified version of the trigonometric distance (G38) with a greater degree of accuracy for points of larger magnitude of values.	1	22
G30	N-gram models [128]	A contiguous sequence of n items from a given sample of text or speech.	1	129
G31	Proportional distance [130]	The sum of squares of the difference between two vectors over the difference between the maximum and minimum values.	1	27

G32	Singular value decomposition [131]	An estimate of where the evolution is going in search-based approaches, by monitoring the movements of individuals across different generations.	1	132
G33	Smith-Waterman distance [133]	The alignment of local sequences for determining similar regions between two strings of nucleic acid sequences or protein sequences.	1	35
G34	Sokal-Sneath distance [134]	A variant of the Jaccard Index (G2) where the weight ω is 2.	1	27
G35	Statistic value X^2 [135]	A distance function that emphasizes large absolute difference existing between the feature values.	1	22
G36	String-Kernels [136]	The inner product between two strings by counting the occurrences of common substrings in the two strings.	1	27
G37	Sellers algorithm [137]	A variation of the edit distance (G3) to find a substring in another string with at most k edit operations.	1	31
G38	Trigonometric distance [127]	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	22
G39	Wasserstein distance [138]	The difference between two frequency distributions over a region, which is also known as the earth mover's distance.	1	116
G40	Word mover's distance [139]	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	47

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

ID	Metric & Source	Description	Total	Papers
S1	Identical transition distance [140]	The number of identical transitions between two finite state machines divided by the average length of paths.	6	140,141,59,35,60,61
S2	Test set diameter (TSDm) [92]	An extension of the pairwise normalised compression distance (G7) to multisets.	5	92,34,15,94,95
S3	Approach level [142]	The number of mismatched branch predicates to reach the target branch.	3	143,144,145
S4	Identical state distance [59]	The number of identical states between two paths of finite state machines divided by their average number of states.	3	59,60,61
S5	Trigger-based distance [59]	An extension of identical transition similarity (S1) to account for triggers in the transitions.	3	59,60,61
S6	Average population diameter [146]	The average distance between all vectors in a population, where the distance between two vectors is the difference of their lengths.	2	146,147
S7	Distinguishing mutation adequacy [148]	An assessment of the diversity of mutants' behaviour based on the mutants' killing information.	2	148,149
S8	Extended subTree distance [94]	A variation of isolated subtree distance (G12) with different mapping conditions.	2	94,95
S9	Path distance [150]	The size of the intersection between the two paths of multisets of trees.	2	94,95

S10	[GUI] State similarity [151]	The difference between the values of two GUI states using the widgets of the GUI.	2	[151,152]
S11	Test diversity [153]	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	[153,154]
S12	[Graph model diversity] Symmetric distance [85]	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	[85]
S13	Text uniqueness [155]	Text matching between two strings, where a string is unique if no other string matches it.	1	[155]
S14	Achieved coverage of pools [156]	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	[156]
S15	Accuracy-based performance measure [157]	The proportion of correctly predicted test inputs to all the test inputs for a DNN.	1	[157]
S16	[Test behavioural similarity] Accuracy (acc) [158]	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	[89]
S17	Average cyclomatic complexity per method (ACCM) [159]	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	[160]
S18	Basic counting [64]	The overlapping occurrences of method calls between two failing sequences of method calls extracted from execution traces of tests.	1	[64]
S19	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]
S20	[GUI similarity] *CONTeSSi*(*n*) [83]	The differences of the frequencies between the past *n* executed events of one test suite to another test suite.	1	[83]
S21	Distance entropy [161]	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	[161]
S22	Enhanced Jaro-Winkler [71]	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	[71]
S23	Graph edit distance [162]	The minimum number of edit operations required to make two graphs identical.	1	[46]
S24	Matthew's correlation coefficient (*MCC*) [163]	A more accurate measure of tests behavioural similarity than accuracy (S16) that accounts for both true positives and true negatives.	1	[89]
S25	Probabilistic type tree [164]	A tree structure to represent a probability distribution over the types.	1	[164]

S26	Response for class (RFC) [159]	The sum of the number of methods inside the class and the number of external methods used by the class.	1	[160]
S27	Syntax-tree similarity [165]	The structural similarity between two sentences represented as “syntax trees” by comparing the tree topologies, node positions, and the types of grammatical relationships.	1	[165]
S28	Traces [134]	The difference between two execution paths, that takes into account the branches covered and the number of times these branches were covered.	1	[166]
S29	Weighted distance function [167]	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	[167]
S30	Extensible access control markup language (XACML) Similarity [168]	The distance between the requests attributes’ values of two XACML test cases and the difference between their policies.	1	[168]

REFERENCES

1. Alpha W. EuclideanDistance - Wolfram: Alpha. 2023.
2. Albunian N, Fraser G, Sudholt D. Measuring and maintaining population diversity in search-based unit test generation. In: 2020:153–168.
3. Arafeen MJ, Do H. Test case prioritization using requirements-based clustering. In: 2013:312–321.
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.
5. Bueno PM, Wong WE, Jino M. Automatic test data generation using particle systems. In: 2008:809–814.
6. Bueno PM, Jino M, Wong WE. Diversity oriented test data generation using metaheuristic search techniques. *Information sciences.* 2014;259:490–509.
7. Carlson R, Do H, Denton A. A clustering approach to improving test case prioritization: An industrial case study. In: 2011:382–391.
8. Chetouane N, Wotawa F, Felbinger H, Nica M. On using k-means clustering for test suite reduction. In: 2020:380–385.
9. Coviello C, Romano S, Scanniello G. An empirical study of inadequate and adequate test suite reduction approaches. In: 2018:1–10.
10. Coviello C, Romano S, Scanniello G, Marchetto A, Antoniol G, Corazza A. Clustering support for inadequate test suite reduction. In: 2018:95–105.
11. Cruciani E, Miranda B, Verdecchia R, Bertolino A. Scalable approaches for test suite reduction. In: 2019:419–429.
12. Farzat Fd. Test case selection method for emergency changes. In: 2010:31–35.
13. Greca R, Miranda B, Gligoric M, Bertolino A. Comparing and combining file-based selection and similarity-based prioritization towards regression test orchestration. In: 2022:115–125.
14. Guarnieri GF, Pizzoleto AV, Ferrari FC. An Automated Framework for Cost Reduction of Mutation Testing Based on Program Similarity. In: 2022:179–188.
15. Haghighatkhah A, Mäntylä M, Oivo M, Kuvaja P. Test prioritization in continuous integration environments. *Journal of Systems and Software.* 2018;146:80–98.
16. Hemmati H, Fang Z, Mantyla MV. Prioritizing manual test cases in traditional and rapid release environments. In: 2015:1–10.
17. Ledru Y, Petrenko A, Boroday S, Mandran N. Prioritizing test cases with string distances. *Automated Software Engineering.* 2012;19(1):65–95.
18. 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.
19. Matinnejad R, Nejati S, Briand LC, Bruckmann T. Effective test suites for mixed discrete-continuous stateflow controllers. In: 2015:84–95.
20. Matinnejad R, Nejati S, Briand LC, Bruckmann T. Automated test suite generation for time-continuous simulink models. In: 2016:595–606.
21. Matinnejad R, Nejati S, Briand LC, Bruckmann T. Test generation and test prioritization for simulink models with dynamic behavior. *Transactions on Software Engineering.* 2019;45(9):919–944.
22. 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.
23. Pei Y, Christi A, Fern X, Groce A, Wong W. Taming a Fuzzer Using Delta Debugging Trails. In: 2014:840–843.
24. 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.
25. Shahbazi A, Miller J. Black-box string test case generation through a multi-objective optimization. *Transactions on Software Engineering.* 2015;42(4):361–378.
26. 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: 2022:564–567.
27. Wang R, Jiang S, Chen D. Similarity-based regression test case prioritization.. In: 2015:358–363.
28. Zhao X, Wang Z, Fan X, Wang Z. A clustering-bayesian network based approach for test case prioritization. In: . 3. 2015:542–547.
29. 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;n/a(37):547–579.

30. 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 (TOSEM).* 2022;31(2):1–36.
31. 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.
32. Oliveira Neto dFG, Ahmad A, Leifler O, Sandahl K, Enoiu E. Improving continuous integration with similarity-based test case selection. In: 2018:39–45.
33. Fischer S, Lopez-Herrejon RE, Ramler R, Egyed A. A preliminary empirical assessment of similarity for combinatorial iteraction testing of software product lines. In: 2016:15–18.
34. Haghighatkhah A, Mäntylä M, Oivo M, Kuvaja P. Test case prioritization using test similarities. In: 2018:243–259.
35. Hemmati H, Briand L. An industrial investigation of similarity measures for model-based test case selection. In: 2010:141–150.
36. Hemmati H, Arcuri A, Briand L. Empirical investigation of the effects of test suite properties on similarity-based test case selection. In: 2011:327–336.
37. 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: 2013:188–197.
38. 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. *Transactions on Software Engineering.* 2014;40(7):650–670.
39. Liu B, Lucia , Nejati S, Briand LC. Improving fault localization for Simulink models using search-based testing and prediction models. In: 2017:359–370.
40. 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.
41. Mariani L, Mohebbi A, Pezzè M, Terragni V. Semantic matching of gui events for test reuse: are we there yet?. In: 2021:177–190.
42. 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.
43. Miranda B, Bertolino A. Social coverage for customized test adequacy and selection criteria. In: 2014:22–28.
44. Miranda B, Cruciani E, Verdecchia R, Bertolino A. FAST approaches to scalable similarity-based test case prioritization. In: 2018:222–232.
45. Oliveira Neto dFG, Feldt R, Erlenhov L, Nunes JBDS. Visualizing test diversity to support test optimisation. In: 2018:149–158.
46. Tang Y, Jiang H, Zhou Z, Li X, Ren Z, Kong W. Detecting compiler warning defects via diversity-guided program mutation. *Transactions on Software Engineering.* 2022;48(11):4411-4432.
47. Viggiato M, Paas D, Buzon C, Bezemer CP. Identifying similar test cases that are specified in natural language. *Transactions on Software Engineering.* 2023;49(3):1027-1043.
48. Xiang Y, Huang H, Li M, Li S, Yang X. Looking for novelty in search-based software product line testing. *Transactions on Software Engineering.* 2021;1(1):1–1.
49. You YS, Huang CY, Peng KL, Hsu CJ. Evaluation and analysis of spectrum-based fault localization with modified similarity coefficients for software debugging. In: 2013:180–189.
50. Zhang Z, Xie X. On the investigation of essential diversities for deep learning testing criteria. In: 2019:394–405.
51. 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.
52. Levenshtein VI. Binary codes capable of correcting deletions, insertions, and reversals. In: . 10. 1966:707–710.
53. Almulla H, Gay G. Generating diverse test suites for Gson through adaptive fitness function selection. In: 2020:246–252.
54. Asoudeh N, Labiche Y. A multi-objective genetic algorithm for generating test suites from extended finite state machines. In: 2013:288–293.
55. Biagiola M, Stocco A, Ricca F, Tonella P. Diversity-based web test generation. In: 2019:142–153.
56. Boussaa M, Barais O, Sunyé G, Baudry B. A novelty search approach for automatic test data generation. In: 2015:40–43.
57. 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.
58. Flemström D, Afzal W, Sundmark D. Exploring test overlap in system integration: An industrial case study. In: 2016:303–312.
59. Hemmati H, Briand L, Arcuri A, Ali S. An enhanced test case selection approach for model-based testing: an industrial case study. In: 2010:267–276.
60. Hemmati H, Arcuri A, Briand L. Reducing the cost of model-based testing through test case diversity. In: 2010:63–78.
61. Hemmati H, Arcuri A, Briand L. Achieving scalable model-based testing through test case diversity. *Transactions on Software Engineering and Methodology (TOSEM).* 2013;22(1):1–42.
62. Ma L, Wu P, Chen TY. Diversity driven adaptive test generation for concurrent data structures. *Information and software technology.* 2018;103:162–173.
63. Mondal D, Hemmati H, Durocher S. Exploring test suite diversification and code coverage in multi-objective test case selection. In: 2015:1–10.
64. Noor TB, Hemmati H. A similarity-based approach for test case prioritization using historical failure data. In: 2015:58–68.
65. Wang W, Wu S, Li Z, Zhao R. Parallel evolutionary test case generation for web applications. *Information and Software Technology.* 2023;155:107113.
66. Wu K, Fang C, Chen Z, Zhao Z. Test case prioritization incorporating ordered sequence of program elements. In: 2012:124–130.
67. 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.
68. Shimmi S, Rahimi M. Leveraging code-test co-evolution patterns for automated test case recommendation. In: 2022:65–76.
69. Hamming RW. Error detecting and error correcting codes. *The Bell system technical journal.* 1950;29(2):147–160.
70. Huang R, Zhou Y, Zong W, Towey D, Chen J. An empirical comparison of similarity measures for abstract test case prioritization. In: . 1. 2017:3–12.
71. 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.
72. Cao Y, Zhou ZQ, Chen TY. On the correlation between the effectiveness of metamorphic relations and dissimilarities of test case executions. In: 2013:153–162.
73. Feng Y, Chen Z, Jones JA, Fang C, Xu B. Test report prioritization to assist crowdsourced testing. In: 2015:225–236.

74. Wang H, Chan W. Weaving context sensitivity into test suite construction. In: 2009:610–614.
75. Wang H, Chan W, Tse T. Improving the effectiveness of testing pervasive software via context diversity. *Transactions on Autonomous and Adaptive Systems (TAAS).* 2014;9(2):1–28.
76. Xie X, Xu B, Shi L, Nie C. Genetic test case generation for path-oriented testing. *Journal of Software.* 2009;20(12):3117–3136.
77. Yoo S, Harman M, Tonella P, Susi A. Clustering test cases to achieve effective and scalable prioritisation incorporating expert knowledge. In: 2009:201–212.
78. Alpha W. ManhattanDistance - Wolfram: Alpha. 2023.
79. Chen J, Wang G, Hao D, Xiong Y, Zhang H, Zhang L. History-guided configuration diversification for compiler test-program generation. In: 2019:305–316.
80. 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: 2021:105–110.
81. Alpha W. CosineDistance - Wolfram: Alpha. 2023.
82. Azizi M. A tag-based recommender system for regression test case prioritization. In: 2021:146–157.
83. Brooks PA, Memon AM. Introducing a test suite similarity metric for event sequence-based test cases. In: 2009:243–252.
84. Lin J, Wang F, Chu P. Using semantic similarity in crawling-based web application testing. In: 2017:138–148.
85. 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.
86. Sondhi D, Jobanputra M, Rani D, Purandare S, Sharma S, Purandare R. Mining Similar Methods for Test Adaptation. *Transactions on Software Engineering.* 2022;48(07):2262-2276.
87. Cilibrasi R, Vitányi PM. Clustering by compression. *IEEE Transactions on Information theory.* 2005;51(4):1523–1545.
88. 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.
89. Oliveira Neto dFG, Dobslaw F, Feldt R. Using mutation testing to measure behavioural test diversity. In: 2020:254–263.
90. Dobslaw F, Oliveira N. dFG, Feldt R. Boundary value exploration for software analysis. In: 2020:346–353.
91. Feldt R, Torkar R, Gorschek T, Afzal W. Searching for cognitively diverse tests: Towards universal test diversity metrics. In: 2008:178–186.
92. Feldt R, Poulding S, Clark D, Yoo S. Test set diameter: Quantifying the diversity of sets of test cases. In: 2016:223–233.
93. 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.
94. 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.
95. Shahbazi A, Panahandeh M, Miller J. Black-box tree test case generation through diversity. *Automated Software Engineering.* 2018;25(3):531–568.
96. Shakhnarovich G, Darrell T, Indyk P. *Nearest-neighbor methods in learning and vision: theory and practice (neural information processing)*. The MIT press, 2006.
97. Mahfoud SW. *Niching methods for genetic algorithms*. University of Illinois at Urbana-Champaign, 1995.
98. Panichella A, Oliveto R, Di Penta M, De Lucia A. Improving multi-objective test case selection by injecting diversity in genetic algorithms. *Transactions on Software Engineering.* 2015;41(4):358–383.
99. Scalabrino S, Grano G, Nucci DD, Oliveto R, Lucia AD. Search-based testing of procedural programs: Iterative single-target or multi-target approach?. In: 2016:64–79.
100. Kulesza A, Taskar B. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning.* 2012;5(2–3):123–286.
101. 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.
102. Gower JC, Legendre P. Metric and Euclidean properties of dissimilarity coefficients. *Journal of classification.* 1986;3:5–48.
103. Tanaka E, Tanaka K. The tree-to-tree editing problem. *International Journal of pattern recognition and artificial intelligence.* 1988;2(02):221–240.
104. Winkler WE. The state of record linkage and current research problems. In: 1999:1–15.
105. Goldreich O. *On testing expansion in bounded-degree graphs*. 20, 2000.
106. Menéndez HD, Jahangirova G, Sarro F, Tonella P, Clark D. Diversifying focused testing for unit testing. *Transactions on Software Engineering and Methodology (TOSEM).* 2021;30(4):1–24.
107. Menendez HD, Clark D. Hashing Fuzzing: Introducing Input Diversity to Improve Crash Detection. *Transactions on Software Engineering.* 2022;48(09):3540-3553.
108. Chandra MP, others . On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India.* 1936;2(1):49–55.
109. 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.
110. Alpha W. CanberraDistance - Wolfram: Alpha. 2023.
111. Alpha W. ChebyshevDistance - Wolfram: Alpha. 2023.
112. Aggarwal CC, Hinneburg A, Keim DA. On the surprising behavior of distance metrics in high dimensional space. In: 2001:420–434.
113. Marculescu B, Feldt R, Torkar R. Using exploration focused techniques to augment search-based software testing: An experimental evaluation. In: 2016:69–79.
114. mathematics oE. Hellinger distance - Encyclopedia of mathematics. 2023.
115. Hellinger E. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen.. *Journal für die reine und angewandte Mathematik.* 1909;1909(136):210–271.
116. Xie X, Yin P, Chen S. Boosting the Revealing of Detected Violations in Deep Learning Testing: A Diversity-Guided Method. In: 2022:1–13.
117. Hill MO. Diversity and evenness: a unifying notation and its consequences. *Ecology.* 1973;54(2):427–432.
118. Nguyen HL, Grunske L. BEDIVFUZZ: Integrating Behavioral Diversity into Generator-based Fuzzing. In: 2022:249-261.
119. 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.
120. Jeffreys H. *The theory of probability*. OuP Oxford, 1998.
121. Menéndez M, Pardo J, Pardo L, Pardo M. The jensen-shannon divergence. *Journal of the Franklin Institute.* 1997;334(2):307–318.

122. Alpha W. Kronecker Delta - Wolfram: Alpha. 2023.
123. Kichigin DY. A method of test-suite reduction for regression integration testing. *Programming and Computer Software.* 2009;35(5):282–290.
124. Kullback S, Leibler R. On information and sufficiencyannals of mathematical statistics. *MathSciNet MATH.* 1951;22(n/a):79–86.
125. Foundation B. Mean squared error - Britannica. 2023.
126. Mosin V, Staron M, Durisic D, Oliveira Neto dFG, Pandey SK, Koppisetty AC. Comparing Input Prioritization Techniques for Testing Deep Learning Algorithms. In: 2022:76–83.
127. Li Z, Hou K, Li H. Similarity measurement based on trigonometric function distance. In: 2006:227–231.
128. Broder AZ, Glassman SC, Manasse MS, Zweig G. Syntactic clustering of the web. *Computer networks and ISDN systems.* 1997;29(8-13):1157–1166.
129. Leveau J, Blanc X, Réveillére L, Falleri J, Rouvoy R. Fostering the Diversity of Exploratory Testing in Web Applications. In: 2020:164-174.
130. Dickinson W, Leon D, Fodgurski A. Finding failures by cluster analysis of execution profiles. In: 2001:339–348.
131. Stewart GW. On the early history of the singular value decomposition. *SIAM review.* 1993;35(4):551–566.
132. Kifetew FM, Panichella A, De Lucia A, Oliveto R, Tonella P. Orthogonal exploration of the search space in evolutionary test case generation. In: 2013:257–267.
133. Smith TF, Waterman MS. Identification of common molecular subsequences. *Journal of molecular biology.* 1981;147(1):195–197.
134. Zalewski M. American fuzzy lop. 2014.
135. Bugatti PH, Traina AJ, Traina Jr C. Assessing the best integration between distance-function and image-feature to answer similarity queries. In: 2008:1225–1230.
136. Leslie C, Eskin E, Weston J, Noble WS. Mismatch string kernels for SVM protein classification. *Advances in neural information processing systems.* 2003;n/a(n/a):1441–1448.
137. Sellers PH. The theory and computation of evolutionary distances: pattern recognition. *Journal of algorithms.* 1980;1(4):359–373.
138. Arjovsky M, Chintala S, Bottou L. Wasserstein generative adversarial networks. In: 2017:214–223.
139. Kusner M, Sun Y, Kolkin N, Weinberger K. From word embeddings to document distances. In: 2015:957–966.
140. Cartaxo EG, Machado PD, Neto FGO. 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.
141. 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.
142. McMinn P. Search-based software test data generation: a survey. *Software testing, Verification and reliability.* 2004;14(2):105–156.
143. 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.
144. Cai G, Su Q, Hu Z. Binary searching iterative algorithm for generating test cases to cover paths. *Applied Soft Computing.* 2021;113:107910.
145. Panchapakesan A, Abielmona R, Petriu E. Dynamic white-box software testing using a recursive hybrid evolutionary strategy/genetic algorithm. In: 2013:2525–2532.
146. Vogel T, Tran C, Grunske L. Does diversity improve the test suite generation for mobile applications?. In: 2019:58–74.
147. 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.
148. Shin D, Yoo S, Bae D. Diversity-aware mutation adequacy criterion for improving fault detection capability. In: 2016:122–131.
149. Shin D, Yoo S, Bae D. A theoretical and empirical study of diversity-aware mutation adequacy criterion. *Transactions on Software Engineering.* 2018;44(10):914–931.
150. Buttler D. A short survey of document structure similarity algorithms. tech. rep., Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States); 2004.
151. Feng J, Yin BB, Cai KY, Yu ZX. 3-way gui test cases generation based on event-wise partitioning. In: 2012:89–97.
152. He Z, Bai C. GUI test case prioritization by state-coverage criterion. In: 2015:18–22.
153. Nikolik B. Test diversity. *Information and Software Technology.* 2006;48(11):1083–1094.
154. Marchetto A, Tonella P. Search-based testing of Ajax web applications. In: 2009:3–12.
155. Alshahwan N, Harman M. Coverage and fault detection of the output-uniqueness test selection criteria. In: 2014:181–192.
156. Yatoh K, Sakamoto K, Ishikawa F, Honiden S. Feedback-controlled random test generation. In: 2015:316–326.
157. 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.
158. Metz CE. Basic principles of ROC analysis. *Seminars in nuclear medicine.* 1978;8(4):283–298.
159. Chidamber SR, Kemerer CF. A metrics suite for object oriented design. *IEEE Transactions on software engineering.* 1994;20(6):476–493.
160. Pizzoleto AV, Ferrari FC, Dallilo LD, Offutt J. SiMut: exploring program similarity to support the cost reduction of mutation testing. In: 2020:264–273.
161. 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.
162. 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.
163. 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.
164. Poulding S, Feldt R. Automated random testing in multiple dispatch languages. In: 2017:333–344.
165. Masuda S, Matsuodani T, Tsuda K. Syntax-Tree Similarity for Test-Case Derivability in Software Requirements. In: 2021:162–172.
166. Reddy S, Lemieux C, Padhye R, Sen K. Quickly generating diverse valid test inputs with reinforcement learning. In: 2020:1410–1421.
167. 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.
168. Bertolino A, Daoudagh S, El Kateb D, et al. Similarity testing for access control. *Information and Software Technology.* 2015;58:355–372.