Survey and Analysis of Quality Measures Used in Aspect Mining

Renata Rand McFadden
Graduate School of Computer and Information Sciences
Nova Southeastern University
Fort Lauderdale, Florida, USA.
randmcfa@nova.edu

Frank J. Mitropoulos
Graduate School of Computer and Information Sciences
Nova Southeastern University
Fort Lauderdale, Florida, USA.
mitrof@nova.edu

Abstract— Aspect mining investigates effective ways of finding crosscutting concerns in existing non-aspect oriented software. These crosscutting concerns can then be refactored into aspects to reduce the system's complexity and make it easier to understand, maintain, and evolve. There have been numerous studies introducing different aspect mining techniques, but they used different quality measures to evaluate their techniques. This paper consolidates a list of these existing quality measurements, discusses how they differ from each other, identifies some examples of how they have been used in previous aspect mining studies, and conducts an analysis of the commonly used metrics for aspect mining clustering. The metrics are compared using real and sample clustering results, identifying their similarities and differences, as well as their strengths and weakness.

Keywords- Aspect Mining; Crosscutting Concerns; Software Metrics; Aspect-Oriented Programming; Quality Measures; Validation

I. INTRODUCTION

Aspect mining investigates effective ways of identifying crosscutting concerns in existing non-aspect oriented software [1]. Once these concerns are identified, the legacy software can be refactored into aspect oriented code, therefore, decreasing its complexity and making it easier to understand, maintain, and evolve [2].

There have been numerous studies researching different aspect mining techniques. However, there has been no standard list of validation measures to evaluate the techniques' effectiveness and to compare the results against other research. As a result, most studies define their own methods to evaluate their technique.

This in turn, makes it difficult to evaluate the effectiveness of these techniques. In order for this area of research to make substantial progress, there needs to be a commonly accepted set of empirical validation methods [1].

In addition, while prior research defined and calculated new metrics, there had been no analysis of how well these metrics represent what they measure and how they compare to each other.

This paper examines the aspect mining literature and consolidates a list of existing quality measurements used across different aspect mining techniques. It then discusses how they differ from each other, the type of aspect mining research they have been used in, and provides examples of research that have used them on sample software.

The paper then presents an analysis of the commonly used metrics for aspect mining clustering and compares the metrics against each other using real and sample clustering results.

The paper is structured as follows. In section 2, we introduce different aspect categories, a glossary of quality measures, and examples of research that used those quality measures. In section 3 we give clustering results for two real clustering algorithms and one sample one. Then for each of the three clustering categories, we identify and calculate metrics, and discuss the results. Section 4 summarizes and discusses our work and Section 5 provides recommendations for further work in this area.

II. ASPECT MINING METRICS

Early aspect mining attempts, for example [3] and [4], did not use any metrics because they were identifying potential aspect candidates that were to be investigated manually. The case studies they conducted were hampered by a lack of benchmarks, which would have all the aspects identified.

Some other studies (e.g. [5], [8], [9], [10], and [20]) manually checked their results for identified aspects and/or false positives, essentially looking for precision and/or recall. However, these studies only investigated some examples of aspects in the source code and did not compute specific metrics.

There are now a few examples of software with documented aspects (e.g. JHotDraw and Laffra's Djikstra algorithm), which allow more recent research to compute metrics and compare results to other studies. These studies are the focus of this paper.

A. Glossary of Quality Measures

ACC (ACCuracy) – metric used for clustered results, measuring accuracy of clustering by calculating the percentage of elements for each crosscutting concern that were discovered in the first cluster where the crosscutting was discovered. A higher value for ACC indicates better clustering results relative to aspect mining [11].

Accuracy – number of correct aspect mining candidates divided by all found candidates [7].

ACT (ACcuracy of Technique) and ACTE (ACcuracy of Technique) – metric used for clustered results, measuring the accuracy of clustering by calculating the percentage of elements for each crosscutting concern that were discovered.

978-1-4799-0053-4/13/$31.00 ©2013 IEEE

This measurement would be used when not all clusters are analyzed. A higher value for ACT indicates better clustering results relative to aspect mining [12, 13].

CODI (Complexity Of DIscovery) – metric used for clustered results, measuring the complexity of crosscutting concerns discovery by measuring the percentage of the elements that need to be analyzed in a partition to discover all crosscutting concerns. All elements that implement crosscutting concerns need to be analyzed. CODI determines how relevant the order in which clusters are analyzed is. A lower value for CODI indicates better clustering results relative to aspect mining, as fewer elements need to be analyzed [14].

CORE (Cohesion Of Recovered crosscutting concern) – metric used for clustered results, measuring the degree to which the crosscutting concerns belong together. A higher value for CORE indicates better clustering results relative to aspect mining [14].

Concern Coverage – Given a total number of concerns, concern coverage is the number of concerns (or concern elements) found divided by the total number of concerns (or concern elements). The total number of concerns (or concern elements) is not necessarily all concerns in the software but just the ones that are known. A higher value for Concern Coverage indicates better result relative to aspect mining [21, 22].

Coverage – proportion of correctly found concern elements over the total number of actual concern elements for that concern. Coverage is based on an individual concern and not all concerns in the system. A higher value for Coverage indicates better results relative to aspect mining [23].

DISP (DISPersion) – metric used for clustered results, measuring the degree of dispersion of crosscutting concerns (how the crosscutting concern methods are spread across the clusters). A higher value for DISP indicates better clustering results relative to aspect mining [15].

DIV (DIVersity) – metric used for clustered results, measuring to which extent each cluster has different crosscutting concerns from other concerns. A higher value for DIV indicates better clustering results relative to aspect mining [15].

DIV2 (DIVersity) – metric used for clustered results, measuring to which extent each cluster has different crosscutting concerns from other concerns. It only sums a single cluster with just non-concern methods in it. This prevents having falsely large value due to many clusters with only non-concern methods in them. A higher value for DIV2 indicates better clustering results relative to aspect mining [16].

False Positive – identifying an element as concern or concern element when it is not in fact a concern [21].

Fscore – measurement that represents the harmonic average of coverage and precision. Fscore = 2/(1/coverage + 1/precision). A higher value for Fscore indicates better accuracy [23].

MTA (Methods To Analyze) – metric used for clustered results, measuring the total number of methods that need to be analyzed in the ordered cluster list to find all methods implementing all concerns. A lower value for MTA indicates better clustering results relative to aspect mining as fewer elements need to be analyzed [16].

PAM (Percentage of Analyzed Methods) - metric used for clustered results, measuring the minimum percentage of methods that need to be analyzed in order to discover all crosscutting concerns for a partition. A crosscutting concern is considered discovered when the first method that implements it is analyzed. A lower value for PAM indicates better clustering results relative to aspect mining as fewer elements need to be analyzed [17].

PAME (Percentage of Analyzed MEthods), PAN (Percentage of ANalyzed methods), and PANE (Percentage of ANalyzed Elements) - metric used for clustered results, measuring the minimum percentage of methods that need to be analyzed in order to discover all crosscutting concerns for a partition. A crosscutting concern is considered discovered when all methods that implement it are analyzed. A lower value for PAM indicates better clustering results relative to aspect mining [11, 12, 15].

PREC (PRECision) – metric used for clustered results, measuring the percentage of found crosscutting concerns. A crosscutting concern is discovered if at least one method is found that implements that concern. A higher value for PREC indicates better clustering results relative to aspect mining [17].

PREC2 (PRECision) – metric used for clustered results, measuring the percentage of all found elements, which implement crosscutting concerns. A higher value for PREC2 indicates better clustering results relative to aspect mining [16].

Precision – the ratio of the actual concern elements versus the ones that have been found by the aspect mining technique [23] or identified by a technique divided by total number of candidates identified by a collection of techniques (basically known crosscutting concerns) [24].

Recall – ratio of correctly found concerns over the total possible concerns. A higher value of recall indicates better results relative to aspect mining [23].

True Positives – computes how many of the found elements are in fact crosscutting concerns, by subtracting non-concern elements found from one and then dividing by all elements found [22].

B. Patterns and Clones

Aspect mining techniques in this category look for patterns, clones, text, type, or signatures of similar elements either in static source code or specifications, and through dynamic program analysis in execution traces. These repetitions of code and/or functionality have been shown to have very high probability of being crosscutting concerns [26]. Most of these studies did not use metrics but essentially checked one or more of the following: concern coverage, precision, or recall.

Table I gives three examples of the studies in this category that computed quality measures.

TABLE I. PATTERNS AND CLONES RESEARCH

Ref	Quality measures	Year	Comments
[6]	Concern coverage, Precision	2004	Clone classes
[7]	Accuracy	2004	Dependence graph and AST
[25]	Precision, Recall	2006	Line co-change

The metrics used for this category follow the information retrieval model [25], where from the found aspect candidates list they measure either how many were real concerns (precision) or what percentage was found from all possible concerns (recall).

The concern coverage is essentially a subset of the recall measurement. It computes percentage of found aspects based on known aspect lists versus all in the system.This is an important distinction and allows the evaluation of the technique's results without knowing all the concerns in the case study software.

Lastly, the Precision measurement as defined by [23] and what [7] called accuracy, are in fact the same metric. They measure how many of the candidates are real aspects versus how many are false positives.

C. *Clustering*

This category has examples of research that uses a clustering algorithm as part of the aspect mining technique. Ideally the clustering results would have a single cluster for each crosscutting concern containing all the elements, such as methods, that implement it. Then it would have one cluster with all the other elements that do not implement any of the concerns.

The quality measures in this category measure either how well the clustering technique groups the aspect mining elements, in other words how close to the ideal, or how long it will take to analyze the clusters to find all of the aspect elements.

Table II gives several examples of the studies in this category that computed quality measures.

TABLE II. CLUSTERING RESEARCH

Ref	Quality measures	Year	Comments
[17]	PREC, PAM	2006	Vector-space model
[12]	ACT, DIV, PAN	2007	Graph-based approach
[13]	ACTE, PREC, DISP, DIV	2008	Vector-space model
[22]	Concern Coverage, True Positives	2008	CBFA
[18]	DISP, DIV	2009	Hierarchical Clustering
[14]	CODI, CORE	2011	Only defined metrics
[16]	DIV2, MTA, PREC2	2012	Vector-space model

For this category, the quality measurements can be divided into three groups. The first group are metrics based on the precision and recall measurements. They either calculate based on finding a concern (e.g. when first element is found or all elements are found) or based on finding one or more elements that implement a concern. The metrics in this category include PREC, PREC2, true positives, false positives, ACT and ACTE.

The next group of metrics in this category computes how well the technique grouped the crosscutting concern elements. The metrics in this category include ACC, DISP, DIV, DIV2, and CORE.

The final group in this category computes how many clusters or methods have to be analyzed to reach some discovery criteria. The metrics in this category include PAM, PAME, PAN, PANE, MTA, and CODI.

All of these metrics are compared and contrasted in the analysis section of the paper.

D. *Other*

This aspect mining category contains examples of research, which used a quality measurement, but did not fit into the other two categories. Like the first category of patterns and clones, most of the research in this category used concern coverage, precison, and recall. Some of the studies also investigated false positives and tried to filter those out as much as possible.

Table III gives several examples of the studies in this category that computed quality measures.

TABLE III. OTHER RESEARCH

Ref	Quality Measures	Year	Comments
[24]	Precision, Recall	2005	Timna
[19]	DIV, PAM, PREC	2006	Graph-based approach
[21]	Concern coverage, False positives	2007	Fan-in, Identifier, Dynamic
[26]	Precision, Recall	2010	Concern Graph
[23]	Precision, Coverage, Fscore	2011	Concern Peers

The research listed in this category, mostly use the same metrics already covered in the previous sections. Most of these metrics are a variation on precision and recall.

Precision measurement, while not discussed earlier, is essentially the same as concern coverage. In [24] the authors tried a number of techniques to find aspects and summed up the aspects to use as the known value. The coverage metric measures the ratio of found to total for an individual concern as opposed to summing and averaging over all concerns, as most of the other metrics do. Lastly, Fscore is harmonic average of coverage and precision, allowing a single value to be used to compare results.

III. ANALYSIS OF ASPECT MINING CLUSTERING METRICS

In an ideal partition, there would be a single cluster for each aspect and an additional single cluster with all the elements not belonging to any aspects. For example, the benchmark JHotDraw, which has 10 identified aspects, would result in a partition with 11 clusters, one cluster for

each aspect and one cluster for non-aspect elements. Metrics for aspect mining measure how well the clustering results compare to this ideal clustering.

A. *Clustering Algorithms*

To analyze the existing clustering metrics, this paper used two clustering algorithms (kmeans and agnes) with a vector-space model clustering approach and previously defined vector model number 6 (fanIn_numCallers_hasMethod_sigTokens), on benchmark JHotDraw [16]. In addition, a sample clustering was defined with significantly better clustering results to allow for additional analysis.

Table IV below gives the clustering distribution for the three algorithms. Each of the clustering results had 530 clusters, a 100% precision, and the results were ordered based on the accumulative fan-in value of each cluster as done by [22, 16].

TABLE IV. CLUSTERING RESULTS

Concern (# methods)	**Clusters KMEANS**	**Clusters AGNES**	**Clusters SAMPLE**
Consistent behavior (21)	15 48 53 55 71 81 110 129 296 303 331 353 443 466	94 136 195 231 259 278 295 320 427 438 481 484 498 500	94 136 150 195 231 259
Decorator (6)	5 31 106 311 507	12 140 184 307	12 140
Composite (12)	125 443 507	136 176 229 307	136 140
Observer (10)	11 36 40 73	26 123 345 361	26 245
Adapter (1)	143	331	331
Command (2)	33 357	221 269	150 192
Contract enforcement (3)	33 273 357	192 221 269	150 192
Persistence (6)	159 208 316	27 245 289	27 35
Undo (3)	33 129 273	192 221 320	150
Exception handling (1)	449	115	115

Comparing the three clustering algorithms' distributions, sample algorithm was designed to have as good or better results and it had 15 clusters with aspect elements in them. The other two clusterings had 32 clusters.

When comparing the number of clusters per aspect in Table IV above, the smaller the value the better the distribution. Ideally each aspect would only have a single cluster. Sample algorithm has one to five clusters per aspect. Agnes algorithm had slightly better results for Decorator aspect while kmeans had slightly better results for Composite aspect. For all other aspects, kmeans and agnes algorithms were the same.

For the overlap or the mixing of aspects in the same cluster, the less mixing of the aspects, the better is the distribution. Both kmeans and agnes algorithms had similar results and sample algorithm only slightly better.

Relative to the number of aspects per cluster, both kmeans and agnes algorithms had the same number of two to three aspects in each cluster. For comparison, sample had the same two to three aspects but significantly less clusters especially for consistent behavior concern.

In the next sections we calculate aspect mining metrics for each clustering category. Then we compare those results with the observations we had made of the clustering distribution.

B. *Precision and Recall Based*

For clustering algorithms, most of them cluster all the elements in the system. That is the case for kmeans and agnes algorithms. Therefore all of the precision and recall related metrics are 100% for the number of found aspects and found aspect elements in a partition. For the purpose of analysis of the metrics, and in fact, the way this would be done when analyzing aspects for refactoring, we set a threshold to only inspect 80% of the clusters in the ordered list. These results are shown in Table V.

TABLE V. PRECISION AND RECALL BASED

Metric	**kmeans**	**agnes**	**sample**
PREC	0.900	0.900	1.000
PREC2	0.985	0.985	1.000
ACT/ACTE	0.900	0.900	1.000
True Positive	0.023	0.022	0.026

PREC and PREC2 are basically recall where PREC considers concern found when single element is encountered and PREC2 calculates for all elements. Sample results were deliberately setup to have 1.0 precision for all metrics in the first few clusters. At 80% (first 424 clusters), both kmeans and agnes found nine out of 10 concerns so PREC is the same at 0.9. At lower percentages of evaluated ordered clusters, agnes was slightly better – at 20% (106 clusters), kmeans had 0.8 and agnes had 0.9 and at 5% (26 clusters) kmeans had 0.5 and agnes had 0.6. While this difference is not necessarily statistically significant, it indicates that agnes algorithm had a slightly better ordering where more aspects were found in earlier clusters.

The PREC metric, as well as the PREC2 metric, when used with a threshold value can be a good indication of ordering as well as the recall. If the difference in PREC values was significant, it would make for a better clustering algorithm as it would require analyzing less clusters and/or methods to find the majority of aspects.

At 80% (first 424 clusters) both kmeans and agnes algorithms were identical 0.985 for PREC2. Even at 20% (first 106 clusters) the results were statistically the same with kmeans 0.769 and agnes 0.785 where agnes had 51 methods and kmeans 50. At 5% (first 26 clusters) it was slightly more

significantly different with kmeans 0.246 (16 methods) and agnes 0.292 (19 methods).

For the clustering methods used, there was not much difference between PREC and PREC2 values as far as comparing the clustering results. When PREC value is significant higher, that clustering algorithm is better at finding different aspects and for ordered clusters; it groups these concerns in the earlier clusters. When PREC2 value is higher, that clustering algorithm finds more elements of aspects but it could be either for the same few aspects or across aspects.

For example, if the algorithm finds many more of the elements of the same aspect but no additional elements for the other aspects, it would signify that it is better in finding that particular aspect only. This could be an interesting finding once it is determined what is different about that particular aspect. Or the algorithm could find more elements across different aspects thus being a more efficient algorithm in general.

The ACT and PREC2 metrics are essentially measuring the same thing. For example they both find that for 80% of ordered clusters there are 64 methods out of 65. However, because of how ACT measures using averages, it loses precision so that while PREC2 gives 0.985 value for kmeans algorithm, ACT gives 0.900.

PREC2 does a straight forward calculation where the number of found methods is divided by the number of all methods that implement a concern. ACT however, does more complicated calculation where for each concern it looks at each cluster and finds percentage of found methods in that cluster that belong to that concern; then it sums these up and divides by the number of concerns.

This is a known issue when calculating using a series of averages. For example, at 20% of ordered clusters, kmeans has 50 out of 65 methods, resulting in PREC2 value of 0.769 and ACT value of 0.633. For agnes algorithm there are 51 out of 65 methods, giving PREC2 value of 0.785 and ACT value of 0.733. A quick glance at the difference in ACT values (0.633 versus 0.733) may give an impression of significant difference but in fact agnes algorithm only found one extra method out of 65 methods.

Both of these calculations are statistically correct and if statistical precision is calculated, they could in fact be statistically the same. Nevertheless, in general, simpler calculations will be more accurate relative to retaining precision and thus could be preferred. If nothing else, when using these metrics, it is important to understand how the calculations are derived and the statistical significance of the results especially when claiming a significantly better result.

True positive for ordered cluster list could be calculated by taking all elements that are concern elements and divide that value by the number of all elements in all the clusters being inspected, in this case the first 80%. The higher value for true positive indicates that the ordered clusters have less non-concern elements in them.

In our experiment, kmeans, agnes, and sample algorithms' true positive values for 80% of clusters were about the same at only 2% of methods being concern elements. As the threshold was decreased however, kmeans and agnes algorithms results stayed basically the same or were worse, while sample algorithm results were better. For example, at 10% (53 clusters) kmeans had 0.033, agnes had 0.027, and sample had 0.093. Then at 1% (5 clusters) kmeans had 0.009, agnes had 0.013, and sample had 0.583.

When these results were viewed along with PREC2 values, it showed that as the threshold was lowered, for kmeans and agnes, the precision decreased, indicating that the concerns were spread into further clusters and had many non-concern elements. Specifically, at 10% kmeans PREC2 value was 0.538, agnes had 0.446, and sample still had 1.0 precision.

So at 10% (53 clusters) sample had significantly better clustering with PREC2 1.0 and True Positives 0.093 (about 701 methods). Then at 1% (5 clusters), sample PREC2 dropped to 0.430 while true positive value had 0.583.

Thus, if the true positive value is low, it means that while the clustering method may have found a significant number of concerns based on PREC2 value, it clustered it with many non-concerns and it would take greater effort to analyze, which are concerns and which are not.

In this way, true positive and PREC2 metrics could be used together to determine a good threshold where a maximized balance is found between precision and true positives, and clustering methods are compared based on that threshold.

C. Grouping Based

Grouping based metrics calculate how well the clustering technique groups aspect elements. For all of these metrics a higher value indicates better clustering result, relative to aspect mining.

The ACC metric calculates the grouping of elements of an aspect within the first cluster that they are found in. As Table VI below shows, all three algorithms for ACC had similar results at about 50%. At first glance it seemed strange because sample algorithm had significantly less number of clusters with concerns and better overall grouping. However, that grouping is not necessarily in the first cluster that the concern is found. For example, for consistent behavior concern, sample algorithm had 21 methods across five clusters unlike kmeans and agnes algorithms which has 21 methods across 14 clusters. Yet, sample algorithm only had one method in the first cluster, while kmeans and agnes had three methods.

This is the obvious weakness of ACC metric. It is possible that the first cluster where some concern element was found is an exception and not the rule and the next cluster with that concern does in fact has most or even all of the elements.

To illustrate, imagine algorithm A that clustered more than one concern element in the same first cluster and then all other concern elements were spread across other clusters. Then imagine algorithm B that had only one concern element in the first cluster but then had all elements of that concern in another cluster. The ACC metric would have a

much higher value for algorithm A based on its definition and yet relative to aspect mining grouping, algorithm B would be a more efficient algorithm. In this extreme case the difference would be in the ordering of the clusters.

TABLE VI. GROUPING BASED

Metric	kmeans	agnes	sample
ACC	0.453	0.486	0.540
DISP	0.435	0.432	0.667
DIV2	0.773	0.778	0.703
CORE	0.295	0.282	0.305

The DISP metric also calculates how well the clustering compares to the ideal of all concern elements being grouped in one cluster. It accounts for all concern elements across all clusters that contain them, thus measuring the degree of dispersion of crosscutting concerns. As expected, the results for kmeans and agnes were essentially the same and much better for sample algorithm. This difference can be seen in the cluster data in Table IV, which shows about the same number of clusters per concern for kmeans and agnes, and much smaller number for sample algorithm.

The DIV2 metric, which measures to what extent each cluster has different crosscutting concerns from other concerns, had about the same results across all algorithms (80%). Inspecting the raw data of mixed clusters, for kmeans and agnes that is consistent but does not seem correct for sample, which had significantly less clusters with multiple concerns.

Closer inspection indicates a weakness in the metric. This metric is based on the number of clusters that have concerns and if the number is significantly different than the final percentage can be misleading, as illustrated in Table VII below.

TABLE VII. DIV2 METRIC BREAKDOWN

breakdown	kmeans	agnes	sample
1 concern only (1.0)	18	19	7
1 concern & non-concern (0.5)	8	7	4
2 concerns only (0.5)	3	1	0
2 concern & non-concern (0.3)	2	4	3
3 concerns only (0.3)	1	1	0
3 concerns & non-concern (0.25)	0	0	1
sum of above rows	24.4	24.5	10.15
methods with concerns	**32**	**32**	**15**

The DIV2 metric would essentially calculate the data shown in Table VII where for example for kmeans row 1, there are 18 clusters with only one concern, giving it a value of 18 x 1.0 = 18 and for row 2, there are 8 clusters with a concern and non-concern thus giving it a value of 8 x 0.5 = 4.0, and so forth. These results are summed up and divided by the number of these clusters.

This way of calculating is an issue where another algorithm, such as sample, has a much smaller number of clusters with concerns, in this case 16 instead of 32. So while the summed up data shows a significant difference between kmeans/agnes and sample algorithm, it has a similar value for the DIV2 metric. In other words, an algorithm which generates a partition with many clusters with a single concern in them, may result in a significantly higher DIV2 then an algorithm, which generates a partition with smaller number of clusters with concerns and better dispersion.

Lastly, the CORE metric had similar results for all algorithms at about 30%. Investigating the data and metric calculations shows that while sample had more concern elements grouped in the same cluster, it also had many non-concern elements in those same clusters. This significantly decreased the cohesion measurement and when averaged across all concern clusters and the number of concerns, it resulted in similar final value. In fact, per concern, sample had better results for five of the 10 concerns; the same for two concerns and worse for three concerns. This breakdown per concern gives a better indication of sample algorithm having a better distribution than the other algorithms, while the final value does not. This again is the known issue with calculations based on a series of averages.

D. *Time for Discovery Based*

Time for discovery based metrics calculate how well the clustering groups the aspects relative to an ordered list of clusters. For all of these metrics a lower value indicates better clustering result, relative to aspect mining.

The PAM metric calculates the number of methods to be analyzed to find all the concerns based on the first found element per a concern. Table VIII shows that kmeans and agnes algorithms had very similar values (30%) and sample algorithm had a much smaller value (3%). When analyzed, these results are reflected in the raw data per concern before averaging is done.

TABLE VIII. TIME FOR DISCOVERY BASED

Metric	kmeans	agnes	sample
PAM	0.303	0.294	0.026
PAME/PAN/ PANE	0.470	0.488	0.043
MTA	0.919	0.948	0.103
CODI	0.380	0.380	0.031

The only difference between PAM and PAME related metrics is the scope. While PAME calculates based on the first concern element, PAME calculates based on all concern elements. As expected, kmeans and agnes algorithms had therefore similar results (50%) and sample algorithm had significantly better results (4%).

PAM and PAME are valid metrics indicating how many clusters would need to be analyzed to find all the concerns

and can be used to compare clustering algorithms. The value, however, is an average of all concerns and clusters and as such not helpful if one wanted to use it on an ordered list to determine how many of those clusters to analyze, for example the first 60% [16].

To illustrate, for the kmeans algorithm, PAME had a value of 47%. However, given an ordered list of clusters, if one analyzed them in order, one would need to analyze 2857 methods out of 3109, which in fact is 92%. In comparison, the sample algorithm would require to analyze 320 methods, which is 10%. And in fact, that is exactly what MTA metric calculates.

For the purpose of comparing the efficiency of an algorithm, PAME accurately shows a significant difference between clustering results of kmeans (47%) versus sample (4%).However, if one is interested in knowing how many methods need to be analyzed in an ordered clustered list, the MTA metric is more appropriate metric to use.

The MTA metric is an alternative to PAME. However, similar logic can be used to define an alternative to PAM where the concern is considered found when first element is found. Also, MTA metric calculates the number of methods that need to be analyzed, however, a similar metric can be defined (for example Cluster To Analyze – CTA) to analyze clusters instead.

Lastly, the CODI metric has similar weakness to PAM and PAME due to the averaging in the calculations. However, this metric definition accounts for permutation sets and so it adds something new. Of course, other metrics can also be used on a number of differently ordered lists and compared to each other but the metric definitions do not explicitly account for that. Having a well-defined definition is beneficial when multiple studies compare their results because there is no ambiguity if their methods compared equivalently.

IV. SUMMARY

To summarize, many of the aspect mining quality measures, across all three aspect mining categories, are based on the precision and recall metrics defined by the information retrieval model. As such, they measure the ratio between real concern(s) or elements of concern(s) over all found concern candidates, as well as the ratio of found concern(s) or elements of concern(s) over all or all known concerns or elements of the concerns. The false positives and true positives are also just derivatives of the precision metric.

The clustering category has some additional quality measures, which either measure the quality of the grouping or the placement of concerns and ordering of the clusters. For the grouping, the measures calculate how well a technique separated the concerns into different clusters or grouped the same concern in the same cluster. The ordering measures compute how well the concerns were placed in the clusters and the clusters were ordered so that the least amount of time is needed to find all concerns.

The analysis of the clustering related metrics showed that while two metrics, such as ACT and PREC2, can essentially measure the same thing, depending on their calculation methods they may have very different results due to a series of averages and therefore a loss of statistical precision. This highlights that when comparing two algorithms based on a metric value, it is important to understand the calculation behind it, and the statistical precision of the results, to determine if a difference in metric value is in fact statistically significant. The CORE metric also has the same statistical precision issue as ACT.

Some metrics when used together, for example PREC2 and True Positives, can give additional insight into the partition distribution without manually analyzing the data. In the analysis illustration for kmeans and agnes algorithm, as threshhold of analysis was decreased, PREC2 value decreased as well, indicating that the concern elements were spread across many clusters further in the ordered list as opposed to the sample algorithm's distribution. Then when the True Positives metric was calculated, it showed that at 10% of analyzed clusters, the precision of sample partition as measured by PREC2 was stil 100% but true positives were only about 9%. Then at 1%, PREC2 value dropped to 43% while true positive value increased to 58%. This indicated that in the earlier clusters, in this case the first 5 clusters, the distribution had decreased number of concern elements but also relatively less non-concern elements.

The analysis also found interesting weaknesses of ACC and DIV2. The ACC metric measures grouping efficiency based on the fist cluster where a concern element is found. If the partition happens to have excellent grouping but not in the first cluster, than the ACC value may be low and misleading for the efficiency of the grouping.

The DIV2 metric's weakness is due to the calculation of average based on the number of clusters that contain concerns. If the number of these clusters is significantly different between two distributions, the final average value may be the same or opposite what one would expect and thus misleading when comparing the diversity efficiency of two partitions.

Lastly, while the PAM and PAME metrics can be used to compare two algorithms, the actual values may be misleading due to the averaging that is done. For example, PAME had a value of 47% for kmeans algorithm results and could be mistakently understood that only 47% of the methods would have to be analyzed in the ordered cluster list. But that is not accurate if one analyzes the clusters sequentially in the ordered list. The reason the value is only 47% is because it is an average across all concerns. If one actually analyzes each element in the partition sequentially, as MTA metric calculated for this scenario, it would require 92% of the elements to be analyzed to find all the concerns.

A similar alternative metric can be used for PAM, as well as variation of MTA, called CTA, to calculate based on the clusters instead of elements.

V. CONCLUSION AND RECOMMENDATIONS

There are a number of quality measures defined and utilized for aspect mining. Many can be used for any aspect mining technique and some are specific to clustering methods. As new aspect mining techniques are compared and contrasted against existing methods, they need to

compute some of the same metrics to properly validate any improvements of the technique. This survey and analysis provides a consolidated list of previously defined and used metrics to assist in selection of the quality measures.

The described analysis of existing quality measures identified some of the metrics' weaknesses and strengths. It illustrated how some of the metrics' results need to be used with caution and understanding of the underlying calculations.

It also demonstrated with the sample distribution, how different views of the data may suggest different results relative to the efficiency of the algorithms being compared. For example, if comparing algorithms with the ACC metric only, one might conclude that all three algorithms cluster about the same. However, using the DISP metric, one would rather conclude that sample is significantly better at aspect mining.

It is always easier to compare methodologies based on a single numeric value. Nevertheless, for clustering methods, as shown in this analysis, it would be advantageous to use a number of different metrics to receive a more accurate view of the results. When a single metric is used, the conclusion needs to be very specific to the characteristics of the data.

Future research could improve on the existing metrics to account for the current weaknesses and also define new ones. It would also be beneficial to suggest a standard list of metrics and combination of metrics to be used across different aspect mining research.

REFERENCES

[1] A. Kellens, K. Mens, and P. Tonella, "A Survey of Automated Code-Level Aspect Mining Technigues," Proceedings of Transactions on Aspect-Oriented Software Development, vol. IV, pp. 143-162, 2007.

[2] M. Marin, A. van Deursen, and L. Moonen, "Identifying Crosscutting Concerns Using Fan-in Analysis," ACM Transactions on Software Engineering and Methodology, vol. 17, pp. 2–37, 2007.

[3] W. G. Griswold, Y. Kato, and J. J. Yuan, "Aspect Browser: Tool Support for Managing Dispersed Aspects," First Workshop on Multi-Dimensional Separation of Concerns in Object-Oriented Systems (OOPSLA '99*)*, pp. 1-6, 1999.

[4] J. Hannemann, and G. Kiczales, "Overcoming the Prevalent Decomposition in Legacy Code," Advanced Separation of Concerns Workshop, International Conference on Software Engineering (ICSE), pp. 1-5, 2001.

[5] T. Tourwe, and K. Mens, K, "Mining Aspectual Views Using Formal Concept Analysis," Proceedings of the Fourth IEEE International Workshop on Source Code Analysis and manipulation (SCAM '04), pp. 97-106, 2004.

[6] M. Bruntink, A. van Deursen, T. Tourwe, and R. van Engelen, "An Evaluation of Clone Detection Techniques for Identifying Crosscutting Concerns," Proceedings of International Conference on Software Maintenance (ICSM 2004), pp. 200-209, 2004.

[7] D. Shepherd, E. Gibson, and L. Pollock, "Design and Evaluation of Automated Aspect Mining Tool," Software Engineering Research and Practice, pp. 601-607, 2004.

[8] D. Shepherd, and L. Pollock, "Interfaces, Aspects, and Views," Linking Aspect Technology and Evolution (LATE) Workshop, pp. 1-6, 2005.

[9] D. Shepherd, T. Tourwe, and L. Pollock, "Using Language Clues to Discover Crosscutting Concerns," Proceedings of the 2005 Workshop on Modeling and Analysis of Concerns in Software, pp. 1-6, 2005.

[10] S. Breu, and J. Krinke, "Aspect Mining Using Event Traces," Proceedings of the 19th International Conference on Automated Software Engineering (ASE'04), pp. 310-315, 2004.

[11] G. Serban, and G. S. Moldovan, "A New Genetic Clustering Based Approach in Aspect Mining," Proceedings of the 8th WSEAS International Conference on Mathematical Methods and Computational Techniques in Electrical Engineering, pp. 135-140, 2006.

[12] G. Serban, and G. S. Cojocar, "A new graph-based approach in aspect mining," proceedins of the International Conference on Knowledge Engineering, pp. 252-260, 2007.

[13] G. S. Cojocar, and G. Czibula, "On Clustering Based Aspect Mining," The 4th International Conference on Intelligent Computer Communication and Processing, pp. 129-136, 2008.

[14] G. Czibula, G. S. Cojocar, and I. G. Czibula, "Evaluation measures for partitioning based aspect mining techniques," International Journal of Computers, Communications, & Control, pp. 72-80, 2011.

[15] G. S. Moldovan, and G. Serban, "A formal model for clustering based aspect mining," Proceedings of the 8th WSEAS International Conference on Mathematical Methods and Computational Techniques in Electrical Engineering, pp. 70–75, 2006.

[16] R. Rand McFadden, and F. J. Mitropoulos, "Aspect-mining using model-based clustering," Proceedings of the 2012 SouthEastConf, pp. 1-8, 2012.

[17] G. Serban, and G. S. Moldovan, "A New k-means Based Clustering Algorithm in Aspect Mining," Eighth International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC'06), pp. 69-74, 2006.

[18] I. G. Czibula I. G. Czibula, G. Czibula, and G. S. Cojocar, "Hierarchical Clustering for Identifying Crosscutting Concerns in Object Oriented Software Systems," In Proceedings of the 4th Balkan Conference in Informatics (BCT 09), pp. 1-8, 2009.

[19] G. Serban, and G. S. Moldovan, "A Graph Algorithm for Identification of Crosscutting Concerns," Informatica, vol. LI, pp. 3-10, 2006.

[20] M. Ceccato, M. Marin, K. Mens, L. Moonen, P. Tonella, and T. Tourwe, "Applying and Combining Three Different Aspect Mining Techniques," Software Quality Journal, vol. 14, pp. 209–231, 2006.

[21] C. K. Roy, M. G. Uddin, B. Roy, and T. R. Dean, "Evaluating Aspect Mining Techniques: A Case Study," Proceedings of 15th IEEE International Conference on Program Comprehension (ICPC '07), pp. 167-176, 2007.

[22] D. Zhang, Y. Guo, and X. Chen, "Automated Aspect Recommendation Through Clustering-Based Fan-in Analysis," International Conference on Automated Software Engineering, pp. 278-287, 2008.

[23] T. T. Nguyen, H. V. Nguyen, H. A. Nguyen, and T. N. Nguyen, "Aspect Recommendation for Evolving Software," Proceeding of the 33rd International Conference on Software Engineering (ICSE '11), pp. 361-370, 2011.

[24] D. Shepherd, J. Palm, L. Pollock, and M. Chu-Carroll, "Timna: A Framework for Automatically Combining Aspect Mining Analyses'" Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering, pp. 184–193, 2005.

[25] G. Canfora, L. Cerulo, and M. Di Penta, "On the Use of Line Co-change for identifying Crosscutting Concern Code," Proceedings of the 22nd IEEE International Conference on Software Maintenance (ICSM '06), pp. 213-222, 2006.

[26] J. Huang, L. Yansheng, and J. Yang, "Aspect Mining Using Link Analysis," 2010 Fifth International Conference on Frontier of Computer Science and Technology, pp. 312-317, 2010.