UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

SOPHOS LTD. and SOPHOS INC.

Petitioners

v.

OPEN TEXT INC.

Patent Owner

Case IPR2023-00528

Patent No. 10,599,844

DECLARATION OF SAM MALEK, Ph.D.
IN SUPPORT OF PATENT OWNER'S PRELIMINARY RESPONSE

TABLE OF CONTENTS

I. QUALIFICATIONS 2

II. UNDERSTANDING OF RELEVANT LEGAL PRINCIPLES 6

III. STATE OF THE PRIOR ART 8

IV. The '844 Patent 11

A. Summary Of The '844 Patent 11

B. Claim Construction 15

C. Level Of Ordinary Skill In The Art 15

V. OVERVIEW OF PRIOR ART 16

A. Overview Of Ideses 16

B. Overview Of Mori 20

VI. Ideses IN VIEW OF mori DOES NOT DISCLOSE OR RENDER OBVIOUS THE CHALLENGED CLAIMS 23

A. Ground 1: Claims 1-20 Were Not Obvious In View Of Ideses 24

1. Ideses does not render Claim 1 obvious 24

2. Ideses does not render claims 2-20 obvious 28

B. Ground 2: Mori Fails To Render Claim 1 Obvious 28

1. Mori fails to render claim obvious because it does not teach or suggest turning features on or off at the classification stage 29

2. Mori fails to render claims 8-10 and 17 obvious 35

C. Ground 3: The Combination Of Mori And Ideses Does Not Render Claims 1-20 Obvious 35

1. Petitioner does not explain the proposed combination. 36

2. Petitioner does not provide a sufficient motivation to combine Ideses and Mori. 36

3. No combination of Ideses and Mori can render the independent claims obvious because neither reference teaches turning features on or off during the classification stage....38

VII. CLAIMS AND/OR LIMITATIONS NOT ADDRESSED IN THIS DECLARATION....38

VIII. SECONDARY OBJECTIVE INDICIA OF NONOBVIOUSNESS 38

IX. CONCLUSION....39

EXHIBIT LIST

Exhibit	Description
2001	Declaration Of Sam Malek, Ph.D.
2002	*Curriculum Vitae* of Sam Malek, Ph.D.
2003	*Webroot, Inc., Open Text Inc., v. AO Kaspersky Lab et al.*, Case No. 6:22-cv-00243-ADA-DTG, Order Granting Third Amended Scheduling Order, Dkt. 142 (W.D. Tex., Dec. 27, 2022)
2004	*Sonrai Memory Ltd. v. Kingston Tech. Co. and Kingston Tech. Corp.,* Case No. 6:21-cv-1284-ADA, Order Denying Kingston Tech. Co. and Kingston Tech. Corp.'s Opposed Motion to Stay Pending Resolution of *Inter Partes* Review, Dkt. 94 (W.D. Tex., Oct. 18, 2022)
2005	*mCom IP, LLC v. Cisco Systems, Inc.*, Case No. 6:22-cv-00261-ADA, Order Denying Defendant's Motion to Stay Pending *Inter Partes* Review, Dkt. 42 (W.D. Tex., Oct. 20, 2022)
2006	*Intellectual Ventures I LLC et al. v. Hewlett Packard Enterprise Co.*, No. 6:21-cv-00596-ADA, Order Denying Hewlett Packard Enterprise Company's Opposed Motion to Stay Pending Resolution of *Inter Partes* Review, Dkt. 104 (W.D. Tex., Dec. 22, 2022)
2007	*Webroot, Inc. and Open Text Inc., v. AO Kaspersky Lab. et al.,* Case No. 6:22-cv-243-ADA-DTG, Claim Construction Order, Dkt. 236 (W.D. Tex. Mar. 16, 2023)
2008	Dani Kass, "Catching Up On Patent Litigation With Judge Albright" (Law360 Mar. 14, 2023); https://www.law360.com/pulse/articles/1582438/print?section=pulse/courts

Exhibit	Description
2009	*Webroot, Inc., Open Text Inc., v. AO Kaspersky Lab et al.*, Case No. 6:22-cv-00243-ADA-DTG, Order Granting Third Amended Scheduling Order, Dkt. 142 (W.D. Tex., Dec. 27, 2022)
2010	United States Courts, Statistics & Reports, Federal Court Management Statistics–Comparison Within Circuit—During the 12-Month Period Ending June 30, 2022, "Comparison of Districts Within the First Circuit — 12-Month Period Ending June 30, 2022," *available* https://www.uscourts.gov/sites/default/files/fcms_na_distcomparison0630.2022_0.pdf
2011	*Webroot, Inc. and Open Text Inc., v. Sophos Ltd.*, Case No. 6:22-cv-00240-ADA, Complaint for Patent Infringement, Dkt. 1 (W.D. Tex., Mar. 4, 2022)
2012	*Webroot, Inc. and Open Text Inc., v. ForcePoint LLC*, Case No. 6:22-cv-00342-ADA-DTG, Defendant ForcePoint LLC's Preliminary Invalidity Contentions (W.D. Tex., Sept. 13, 2022)
2013	*Webroot, Inc. and Open Text Inc., v. Sophos Ltd.*, Case No. 6:22-cv-00240-ADA-DTG, Defendant Sophos Ltd.'s Preliminary Invalidity Contentions (W.D. Tex., Sept. 13, 2022)
2014	*Webroot, Inc. and Open Text Inc., v. Trend Micro Inc.*, Case No. 6:22-cv-00239-ADA-DTG, Defendant's Preliminary Invalidity Contentions (W.D. Tex., Sept. 13, 2022)
2015	*Webroot, Inc. and Open Text Inc., v. Ao Kaspersky Lab*, Case No. 6:22-cv-00243-ADA-DTG, Defendant's Preliminary Invalidity Contentions (W.D. Tex., Sept. 14, 2022)
2016	*Webroot, Inc. and Open Text Inc., v. CrowdStrike, Inc. and CrowdStrike Holdings, Inc.*, Case No. 6:22-cv-00241-ADA-DTG, Defendant's Preliminary Invalidity Contentions (W.D. Tex., Sept. 13, 2022)

I, Sam Malek, Ph.D., declare as follows:

1. I have been engaged by King & Spalding LLP on behalf of OpenText Inc. ("Patent Owner"), to provide this Declaration concerning the technical subject matter relevant to the petition for Inter Partes Review ("Petition") of U.S. Patent 10,599,844 ("the '844 patent").

2. I am over 18 years of age. I have personal knowledge of the facts stated in this Declaration and could testify competently to them if asked to do so.

3. I have reviewed and am familiar with the specification and the claims of the '844 patent. In general, I will cite to the specification of a United States patent using the following formats: (Exhibit No. Col:Line Number(s)) or (Exhibit No. Paragraph Number(s)). For example, the citation (EX1001 1:1–10) points to the '844 patent specification at column 1, lines 1–10. Also, for convenience, I use italics to denote limitations from the challenged claims.

4. All the opinions in this Declaration are based on the documents I reviewed and my knowledge, experience, training and professional judgment. In forming the opinions expressed in this Declaration, I reviewed the Petition for Inter Partes Review of U.S. Patent No. 10,599,844 and supporting exhibits.

5. I have been asked to provide my technical opinions regarding how a person of ordinary skill in the art would have understood the claims of the '844 patent at the time of the alleged invention. I have also been asked to provide my

technical opinions on how concepts in the '844 patent specification relate to claim limitations of the '844 patent. In reaching the opinions provided herein, I have considered the '844 patent, its prosecution history, and the references cited herein and have drawn as appropriate on my own education, training, research, knowledge, and personal and professional experience.

I. QUALIFICATIONS

6. In formulating my opinions, I have relied on my knowledge, training, and experience in the relevant filed, which I will summarize briefly. A more detailed summary of my background, experience, and publications is set forth in my curriculum vitae (CV), which is submitted as Exhibit 2002.

7. I have more than two decades of experience in the general area of computer and software technology in a variety of roles, including as a software engineer, software architect, and programmer at a variety of companies, as well as a researcher and professor at several universities.

8. I am currently a Full Professor of Software Engineering with tenure in the Informatics Department within the School of Information and Computer Sciences at the University of California, Irvine. I am also the Director of the Software Engineering and Analysis Laboratory ("SEAL"). My general research interests are in the field of software engineering, and to date my focus has spanned the areas of software analysis and testing, mobile computing, architecture and design, and

software dependability and security. The underlying theme of my research has been to devise techniques and tools that aid with the construction, analysis, and maintenance of largescale software systems.

9. I received my Ph.D. and M.S. degrees in Computer Science from the University of Southern California and my B.S. degree in Information and Computer Science from the University of California, Irvine. I have received numerous awards for my research contributions, including the Test of Time Award from the Association for Computing Machinery Special Interest Group on Software Engineering (2020), National Science Foundation CAREER award (2013), GMU Emerging Researcher/Scholar/Creator award (2013), and the GMU Computer Science Department Outstanding Faculty Research Award (2011).

10. I have authored over 120 peerreviewed research publications. These publications have been highly cited (close to 10,000 times according to scholar.google.com). I have received numerous research grants from a variety of organizations, including the National Science Foundation (NSF), Federal Bureau of Investigation (FBI), Air Force Office of Scientific Research, Department of Homeland Security (DHS), National Security Agency (NSA), Defense Advanced Research Projects Agency (DARPA), Office of the Director of National Intelligence, and Army Research Office among others. Software systems and results produced in

my research lab have been adopted for use by various organizations in both private (e.g., Boeing, Bosch) and public (e.g., NASA, DHS, FBI) organizations.

11. I also have several years of experience as a software engineer, software architect, and programmer at a variety of companies, namely the Boeing Company, PricewaterhouseCoopers Consulting (later acquired by International Business Machines ("IBM") Corporation), FieldCentrix, and Neural Computing Systems Lab. During my tenure in the industry, I gained professional experience in the design, development, and management of largescale software engineering efforts. Among the software systems that I designed and developed were commercial ecommerce and enterprise software systems. An important part of my responsibility in these projects was implementation and evaluation of security mechanisms to protect the resulting software systems from security attacks.

12. I have taught software engineering and computer science courses on a regular basis at the B.S., M.S., and Ph.D. levels at the universities where I have been employed. I have been a frequent speaker at scientific conferences, symposia, workshops, and gatherings of software engineering academics and professionals. I have served as chair, committee member, and reviewer for numerous software engineering journals, magazines, and conferences. I am currently on the editorial board of the ACM Transactions on Software Engineering and Methodology, the ACM Transactions on Autonomous and Adaptive Systems, and the Springer Journal of

Computing. I am a member of the Association for Computing Machinery (ACM), ACM Special Interest Group on Software Engineering (SIGSOFT), and the Institute of Electrical and Electronics Engineers (IEEE). I have testified at trial and deposition as an expert witness in numerous cases in the past five years.

13. My experience is highly relevant to the technologies at issue in this case and their application here. As set forth above, during my tenure in industry, I contributed to the design and development of several largescale enterprise and ecommerce software systems. In most of these systems, security and integrity of the system was of great concern. I contributed to the design and construction of proper security methodologies, such as the implementation of access control policies, in these systems. After this experience, as part of my research at the university over the past 15 years, I have conducted original research and published peerreviewed manuscripts related to software security. Some of this work has involved the use of static and dynamic program analysis techniques together with machine learning for detection of security threats. For example, in a paper titled "Lightweight, ObfuscationResilient Detection and Family Identification of Android Malware," published in the ACM Transactions on Software Engineering and Methodology in 2018, I described a tool called RevealDroid that statically extracts information from Android apps to train a machine learning model to distinguish between malware and benign apps. RevealDroid was adopted and used by the Department of Homeland Security

in their Software Assurance Marketplace platform for vetting software products used in the government. My work in this area was sponsored by more than 2 million dollars in research funding from the Department of Homeland Security, the National Security Agency, the Office of Director of National Intelligence, Defense Advanced Research Projects Agency, and the Federal Bureau of Investigation.

14. As part of the courses that I regularly teach at the University at both undergraduate and graduate levels, I cover topics that are directly related to the technologies at issue in this case including software security, security vulnerabilities and exploits, program analysis, artificial intelligence, and machine learning. As associate editor of several journals and committee member of many conferences, I regularly evaluate the novelty of article submissions related to the technologies at issue in this case (e.g., software security, program analysis, machine learning) for publication in these venues.

II. UNDERSTANDING OF RELEVANT LEGAL PRINCIPLES

15. I am not a lawyer, and I will not provide any legal opinions. Although I am not a lawyer, I have been advised certain legal standards are to be applied by technical experts in forming opinions regarding the meaning and validity of patent claims.

16. I understand that a patent claim is invalid if it is anticipated or obvious in view of the prior art, and that a claim can be unpatentable even if all of the

requirements of the claim cannot be found in a single priorart reference. I further understand that invalidity of a claim requires that the claim be anticipated or obvious from the perspective of a person of ordinary skill in the art at the time the invention was made.

17. I have been informed that a patent claim is invalid if it would have been obvious to a person of ordinary skill in the art. In analyzing the obviousness of a claim, I understand the following factors may be taken into account: (1) the scope and content of the prior art; (2) the differences between the prior art and the claims; (3) the level of ordinary skill in the art; and (4) any so called "secondary considerations" of nonobviousness if they are present.

18. I have been informed that a claim can be obvious in light of a single prior art reference or multiple prior art references. I understand that to prove that prior art or a combination of prior art renders a patent obvious, it is necessary to:

1) Identify the particular references that, singly or in combination, make the patent obvious;
2) Specifically identify which elements of the patent claim appear in each of the asserted references; and
3) Explain why a person of ordinary skill in the art would have combined the references, and how they would have done so, to create the inventions claimed in the patent. I further understand that exemplary rationales that may support a conclusion of obviousness include:

- Combining prior art elements according to known methods to yield predictable results;
- Simple substitution of known element for another to obtain predictable results;
- Use of known technique(s) to improve similar devices (methods or products) in the same way;
- Applying a known technique to a known device (method or product) ready for improvement to yield predictable results;
- "Obvious to try" – choosing from a finite number of identified, predictable solutions with a reasonable expectation of success;
- Known work in one field of endeavor may prompt variations of the work for use in either the same field or a different field based on design incentives or other market forces if the variations are predicable to a person of ordinary skill in the art; and
- Some teaching, suggestion, or motivation in the prior art that would have led a person of ordinary skill in the art to modify the prior art reference or to combine the prior art reference teachings to arrive at the claimed invention.

19. I have been informed that, in considering obviousness, hindsight reasoning derived from the patentatissue may not be used.

III. STATE OF THE PRIOR ART

20. At the time of the '844 patent, conventional approaches for diagnosing potential malware threats were costly and time consuming, making it difficult to

realistically address zero-day threats for all of the files entering a system. One of these approaches, which may be described as "static code-based analysis," "typically focus[ed] on finding [known] malicious code blocks within a file." (EX-1001 2:15-17). The drawbacks of codebased analysis are twofold: first, because it could only identify code blocks in an executable that were already known to be malicious, it could not identify a new threat; second, this approach was slow and resource intensive, requiring substantial investments in time and computational resources to disassemble the code and, if the executable was encrypted, to decrypt it as well.

21. One potential solution to the first problem (i.e., identifying malicious code that had not been seen before) is to use a "machine learning classifier." (EX-1001 1:20-40.) Machine learning classifiers are software programs that are "trained" to recognize, and classify, the malicious or benign nature of new, unknown executables. They do so by training to recognize certain indicia of maliciousness or benignness, known as "features," within a large group of executables whose degree of maliciousness is already known (*e.g.*, "known malicious executable files, known benign executable files, and [known] potentially unwanted executable files"). (*Id.*) This process is known as the "training" phase.

22. Based on these identified features, machine learning systems build a "classifier"—i.e., a predictive model that accepts a set of features as input and outputs a predicted classification. Once trained, this model can be used to classify

previously unseen executables: the security software extracts from the previously unseen executable the features that the classifier has been trained to recognize, inputs those features into the classifier, and obtains the predicted classification for the executable. These are the steps of the "classification stage."

23. One exemplary machine learning classifier is known as a support vector machine ("SVM"). A SVM is typically trained by extracting a set of features (represented as a fixed-length "feature vector") from each training sample and then finding a mathematical function that best separates the feature vectors into respective categories. To classify a new sample, the fixed set of features is extracted from the new sample (a process known as "feature extraction"), the features are used to create an input feature vector with the number of elements required by the SVM, and the feature vector is input into the SVM for classification. Other machine learning classifiers, such as artificial neural networks, process feature vectors similarly to obtain a resultant classification.

24. Because the computational time and resources required grows with the number of features a classifier has been trained to consider, the training phase sometimes includes a process known as "feature selection," which is synonymous with the phrase "feature pruning." Feature selection is the process of identifying the features that are the most empirically relevant to a determination of maliciousness, and configuring the classifier to consider only those features. Feature selection at the

training stage reduces the computational burden of the classification stage, as the classifier will need to extract fewer features and/or perform less complex mathematical computations.

IV. THE '844 PATENT

A. Summary Of The '844 Patent

25. The '844 patent improves upon the prior art by introducing a technique of performing feature selection at the classification stage (rather than merely during the training stage). Performing feature selection during the classification stage allows the system to dynamically calibrate its performance based on the content of the executable, turning "off" features that are unlikely to be valuable for classification of this executable. Doing so accelerates the performance of the classifier by not forcing it to determine a value for features unlikely to facilitate a positive categorization of the executable. It thus improves upon prior art "static" evaluation, whereby the classifier had to always evaluate, or "test," every feature predetermined during the training stage.

26. In the "training stage," the '844 patent discloses a system that examines a training set of executables known to be malicious, benign, or potentially unwanted, and identifies a combination of features within those executables that function as indicia of the category with which they are correlated (i.e. features designating an executable as malicious, benign, or potentially unwanted). Using these identified

feature sets, the system generates a "classifier:" a predictive model that extracts features from a new, previously unknown executable, and outputs a category for that executable.

27. The '844 patent achieves this result by turning features "on" or "off" during the classification stage based on the content of the executable being evaluated. (EX-1001 7:57-8:22; 12:54-13:4.)

28. For example, the classifier may consider certain features (at the classification stage) only if the size of the executable being evaluated is above a given threshold or only if the executable includes a certain type of content (e.g., legal copyright information), or a particular permutation of that content's subtypes (e.g. for legal information, "timestamp data," "licensing information," "copyright information,"). (*Id.* 7:57-8:22.)

29. To decide which features to turn on or off, the '844 Patent teaches analyzing the values of the "static data points" in the executable. A "static data point" may be a "point of reference," such as the executable's file size, section names, legal information, and so forth. (*See* EX-1001 5:62-6:58, 8:53-56.)

30. The specification provides many exemplary embodiments of static data points (EX-1001 Table 1.1, 8:53-56), and a POSITA would recognize that many of these embodiments correspond to fields defined by the "Portable Executable" file format, such as "file size," "PE header time," and "linker version." (*See* EX-1001

3:35-46, 4:22-24 (describing extraction of static data points from PE files). In such embodiments, a static data point may include a field name (i.e., a predefined character string such as "File Size") and a field value (*e.g.*, a number such as 4982784).) (EX-1001 8:53.)

31. The claimed system can turn features on or off during the classification phase based on the values of the static data points extracted from the executable being classified. (EX-1001 5:62-6:58, 8:53-56.) For example, if the value of the "File Size" static data point is greater than a certain numerical threshold, then the classifier may determine values for fewer or additional features, in order to make a positive decision about whether the executable is malicious, benign or potentially unwanted. (*Id*.) The set of features ultimately extracted from the executable being classified may be used to create one or more feature vectors that are input into the appropriate classifier. (*Id.* 8:49-9:40.)

32. Claim 1 recites a method that corresponds to the classification stage of the invented technique:

> A computer-implemented method comprising:
>
> extracting a plurality of static data points from an executable file without decrypting or unpacking the executable file, wherein the plurality of statc data points represent predefined character strings in the executable file;
>
> generating a feature vector from the plurality of static data points using a classifier trained to classify the plurality of static data points based on a collection of data comprising known malicious executable files, known benign executable files, and known unwanted executable files,

> wherein the collection of data comprises at least a portion of the plurality of static data points, and
>
> wherein one or more features of the feature vector are selectively turned on or off based on whether a value of one or more static data points from the plurality of extracted static data points is within a predetermined range;
>
> and evaluating the feature vector using support vector processing to determine whether the executable file is harmful.

(EX-1001 (claim 1).)

33. The claimed method begins with extracting static data points from the executable being classified. As described above, these data points represent predefined character strings in the executable (*e.g.*, a "File Size" field of a PE file). The method then invokes a machine learning "classifier" that was previously trained on "known malicious," "known benign," and "known unwanted" executables that included at least some of the same static data points. This classifier generates a feature vector from the static data points it extracted from the executable, and selectively turns certain features on or off if the values of those static data points are within certain predetermined ranges. Thus, as explained above, the classifier might turn certain features "off" if the "File Size" of the executable is smaller than some predefined threshold. Finally, the classifier classifies the executable by using support vector processing to evaluate the resultant feature vector.

34. Claim 17 recites a system that performs a variation of the method recited in claim 1: the system of claim 17 extracts predefined character strings (rather than static data points), uses those to create a feature vector, and turns certain features on or off based on the values of those predefined character strings.

B. Claim Construction

35. I understand that the the parties have agreed that no claim terms need be construed for this Petition, and that all claim terms should carry their plain and ordinary meaning in light of the intrinsic record. (Petition 7-10.)

36. Consistent with that understanding, I apply the plain and ordinary meaning of all claim terms.

C. Level Of Ordinary Skill In The Art

37. I understand that certain issues relating to the validity of the '844 patent must be judged from the perspective of a person of ordinary skill in the relevant art, as I discuss below. I have been asked to define the level of a "person having ordinary skill in the art" or "POSITA" that someone would have had at the time the alleged invention as claimed was made. In deciding the level of ordinary skill, I have considered the following factors, which I have been informed are relevant in determining the level of ordinary skill in the art:

- The levels of education and experience of persons working in the field;
- The types of problems encountered in the art;

- The prior art solutions to these problems;
- The rapidity with which innovations are made; and
- The sophistication of the technology.

38. I understand that Petitioner has defined the level of ordinary skill in the art as:

> A person of ordinary skill in the art ("POSITA") on May 12, 2015, would have a Bachelor's degree in Computer Science, Computer Engineering, Electrical Engineering, or an equivalent discipline, and at least one year's worth of experience relating to static data analysis and malware detection. In the alternative, a POSITA could have two or more years of work experience relating to static data analysis and malware detection in lieu of the education requirements.

(Petition 11.)

39. I agree with the Petitioner's definition of a POSITA.

40. I was at least a person of ordinary skill in the art at the time of the invention of the '844 patent.

V. OVERVIEW OF PRIOR ART

A. Overview Of Ideses

41. Ideses is directed to the generation of machine learning classifiers that are suitable for resource-constrained client devices, such as smartphones. EX-1005 at [002]-[0006].

42. Because of the computational complexity of the operations involved in machine learning classification and training, mobile devices, which are often "resource limited," Ideses discloses leveraging a "central server" to perform a substantial portion of such operations. (EX-1005 [0060].) In particular, Ideses discloses using this server to identify, at the training stage, an optimal, "pruned" set of features that are both computationally inexpensive to test and empirically effective at categorizing a given application as desirable or undesirable. (*Id.* [0061].) After this pruned set of features has been introduced into the mobile device, "classification…may take place" on that device "during run-time" within the resource constraints of the device. *Id*.

43. For the sake of clarity, I have reproduced Figs. 1 and 2 of Ideses, which respectively demonstrate the "training stage" (Fig. 1) and the "classification stage" (Fig. 2) disclosed by Ideses, with an annotation (in RED) indicating at which stage feature selection occurs.

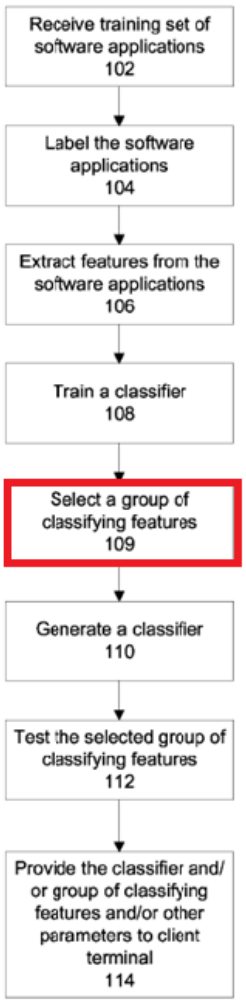


Fig. 1 (Training)

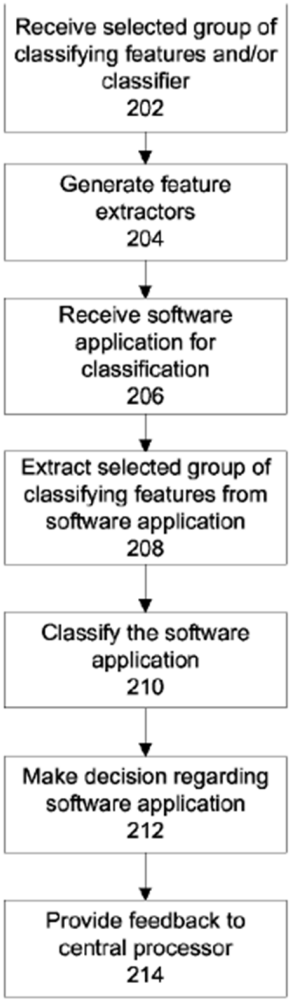


Fig. 2 (Classification)

(EX-1005 Figs. 1 and 2 (annotated).)

44. As shown in Fig. 1 above, feature selection in Ideses occurs at the training stage. This stage, as disclosed by Ideses, begins with the central server receiving a "training set of software applications" (step 102). That training set is then categorized (step 104), i.e. "Labeled," as either "desired" or "undesired." (EX-1005 [0081]-[0085].) This labeling can be performed either by human supervisors or through a "non-supervised", *e.g.* algorithmic approach, that relies on drawing

correlations between clusters of applications to determine which category each application belongs to. (*Id.* [0085].) In the next step (step 106), features are extracted from each categorized software application, and then (step 108) those features (and the categories with which they are associated) are used to train the classifier to recognize an application as desired or undesired. (*Id.* [0094]-[0096].)

45. At step 109, The training system disclosed by Ideses performs feature selection and feature "pruning"—i.e., the process of selecting the features that the trained classifier will consider (step 109). Ideses teaches that choosing the appropriate features most relevant to classification is based on two factors:1) "identifying a classification effectiveness rank for each of the features" and/or 2) "resource requirements for each of the features." (EX-1005 [0099] (emphasis added).) In other words, features are selected at the training stage by optimizing "the tradeoff" between "improving classification ability" and "reducing CPU requirements and/or … memory usage." (*Id.* [0113].)

46. By performing this optimization, "the group of classifying features is selected" to both achieve accurate classification and "to allow calculation and/or extraction of the features during run-time on the mobile device." (*Id.* [0103].)

47. Figure 2 illustrates the classification stage disclosed by Idseses, as executed by a resource-constrained mobile device—i.e., "execution of a classifier based on a pruned feature list" that the mobile device can both extract and classify

using its own resources. (EX-1005 [0123].) The mobile device: (1) receives the classifier and pruned feature list from the server (step 202); (2) generates or identifies a feature extractor for the pruned feature list (step 204); (3) receives a new executable to evaluate (step 206); (4) uses the extractors to extract the pruned feature list from the new executable (step 208); (5) inputs the extracted features into the classifier to classify the new executable (step 210); uses the output of the classifier to decide whether to execute the new executable (step 212); and (6) reports the outcome back to the server (step 214).

B. Overview Of Mori

48. As with Ideses, Mori discloses a technique of training, a machine learning classifier to identify malware. Mori leverages "word extraction" to identify "readable strings" within an executable: i.e. words within an executable that appear within a dictionary or are retrieved with a search engine. (EX-1006 [0023], [0032], [0066].)

49. Almost the entirety of Mori is devoted to a description of its classifier's "supervised machine learning system," i.e. of its "training stage." (EX-1006 [0031], [0055]-[0060]) (teaching that "each 'malware' and 'normal' file feature vector is learned in advance"), *id.* [0061] (describing feature vectors, in Table 5, for files whose "class," i.e. whether it is "malware" or "normal," is already known). This "supervised machine learning system" consists of the following steps: 1)first, the

system receives a training set of "objects," or executables; 2) second, it creates a feature vector for each "object"; and 3) finally, it uses the collection of feature vectors to train a classifier (*e.g.*, an SVM) for classifying new executables. (*Id.* [0010], [0029]-[0031].)

50. The system begins by identifying all readable words that appear in the executables, or "objects," in the training set. (EX-1006 [0030], [0052]-[0053]) (teaching that the feature vectors are created "[u]sing each word … in each file"). The system then creates a feature vector for each executable. The feature vectors each represent M features, where each feature corresponds to a respective one of the readable words identified across all executables in the training set. (*Id.*) For example, if Table 3 of Mori represents each readable string found within a given executable, the supervised machine learning system described by Mori would create a feature vector comprised of seven values—one for each of the readable words in the table. (EX-1006 [0040].)

51. For each executable within the training set, the system described by Mori sets the value of each of the M features in the executable's feature vector to reflect the occurrence, frequency, or importance of the readable word corresponding to that feature. (EX-1006 [0030].) For example, the value of a given feature could be set to 1 or 0 based on "whether a word [corresponding to that feature] existed [in the executable] or not." (*Id.*) Alternatively, the value of a feature could reflect the

"frequency" or "normalized frequency" with which the corresponding word appears in the executable or a "tf-idf value" for the word (i.e., the word's frequency in the executable compared to its frequency in the other executables in the training set). (*Id.*)

52. In Mori, a feature's value will be set to "0" if the readable word corresponding to that feature does not appear within the executable. For instance, when the value is set to 1 or 0 based on "whether a word existed," the absence of a word in a given executable is denoted by ascribing the value "0" to the feature that corresponds to that word within the feature vector. (EX-1006 [0030].)

53. Similarly, embodiments of Mori where the value of each feature is a tf-idf value (which represents the "frequency" of the corresponding word within the executable), the absence of the corresponding word in an executable would likewise be noted with a value of 0 for the corresponding feature in the executable's feature vector, "because the tf portion is 0." (Petition ¶ 161.) Therefore, in all embodiments of Mori, the absence of a given readable word from an executable is signified in the executable's feature vector by setting the value of the feature corresponding to that readable word to "0."

54. Once Mori's system has created a feature vector for each executable within the the training set, those feature vectors are used to train a classifier (e.g.,

SVM), and that classifier can be used at the classification stage to classify new executables.

55. Mori describes the classification stage only briefly. (*See* Ex.1006 [0054]-[0055]); *see also*[0034].) Here, Mori merely discloses analyzing an executable to determine values for the M features defined in the training stage—i.e., the occurrence or frequency with which the M words predefined in the training stage appear in the executable. The system places those values in a feature vector for the executable and inputs that feature vector into the classifier for classification. (*Id.* at [0054].) Importantly, the feature vectors employed at this classification stage are static rather than calibrated on the basis of the content of the executable to be classified: i.e. the same set of M features that were defined for a given feature vector in the "training stage" is the set that is extracted and evaluated at "classification."

VI. IDESES IN VIEW OF MORI DOES NOT DISCLOSE OR RENDER OBVIOUS THE CHALLENGED CLAIMS

56. I understand that The Petitioner and its expert, Dr. Gene Sudik, allege that Ides and Mori, both individually and in combination, render obvious every limitation of independent claims 1, 10 and 17, and this allegation underlies their challenge of the remaining dependent claims 2-9, 11-17 and 18-20.

Ground 1	Section 103(a)	Obvious over Ideses	Claims 1-20

Ground 2	Section 103(a)	Obvious over Mori	Claims 1-20
Ground 3	Section 103(a)	Obvious over the combination of Ideses and Mori	Claims 1-20

57. I disagree with the arguments in the Petition and the opinions set forth by Dr. Tsudik for at least the reasons that I discuss in further detail below.

A. Ground 1: Claims 1-20 Were Not Obvious In View Of Ideses

58. In my opinion, Ideses does not render obvious independent claims 1, 10 and 17. Therefore, it is also my opinion that Ideses does not render any of the dependent claims obvious.

59. As I explain above, Ideses merely discloses "pruning" or selecting features ***at the training stage*** based on 1) how computationally expensive they are to extract and evaluate; and 2) empirical determinations of how effective these features are at making a positive categorization of a given application as "desired" or "undesired."

60. In stark contradistinction to Ideses, the '844 patent discloses selectively turning features on or off at 1) the classification stage, 2) based on the value of the static data points that have been extracted from an executable. It is thus my opinion that none of the claims of the '844 patent are rendered obvious by Ideses.

1. **Ideses does not render Claim 1 obvious**

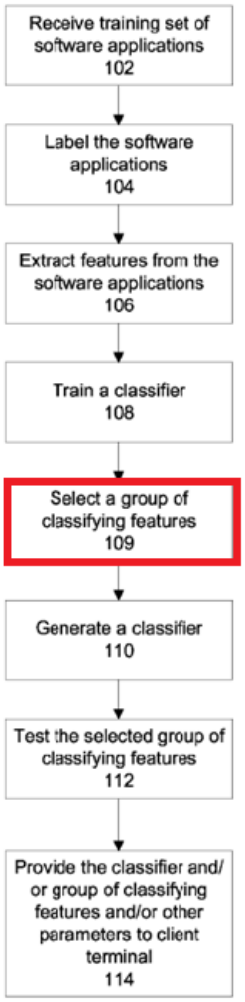
61. As I explain above in Part IV.A, Claim 1 of the '844 patent is directed to feature selection at the classification rather than the training stage, and that

selection (i.e. turning features "on" or "off") is performed based on the extracted content (i.e. the values of the "static data points") of the executable being evaluated. (EX-1001 7:57-8:22; 12:54-13:4.) In my opinion, Ideses does not disclose the invention recited in Claim 1 because, in Ideses: 1) feature "pruning" occurs while the classifier is being "trained," and the set of features that will be used to classify an application is determined at the training stage; and 2) the features included in that set will not vary at the classification stage, based on content extracted from the application being classified.

62. In my opinion, the teachings of Ideses relied upon by the Petition are unambiguously directed to performing selection at the "training" stage, based on the features' effectiveness and computational expense, rather than at the classification stage, based on the content of the executable being classified.

63. Both Petitioner and Dr. Tsudik point to "the method depicted in Figure 1" of Ideses as disclosing feature selection at the classification stage. (Petition 30; EX-1004 at 40 ("Ideses explains that in the method depicted in Figure 1, 'the full feature list is optionally selectively pruned, that is, features are 'selectively turned off' as required by limitation [1.b].").

64. That is erroneous, because Figure 1 is clearly directed to ***the training*** stage for the classifier.



(EX-1005, Fig. 1 (Training), annotated.)

65. As depicted above, Figure 1 of Ideses describes the training stage: the system training on a received set of "labeled" applications (steps 102 and 104) and ultimately "generating" the classifier (step 110) and providing the classifier to the mobile device (step 114). This all occurs before the classification stage. Indeed, Ideses discloses that "the pruned feature set" determined in the training stage (Step 109) is "provided to the mobile device for performing local run-time feature extraction and classification" (Step 114), and thus the pruned feature list must exist *before*

the classification stage described in Figure 2. (EX-1005 [0061]; *see also* , Fig. 4 (conflating the "training" and "pruning" module).)

66. Because the feature selection depicted in Figure 1 occurs before Ideses' system even receives the software application to be classified, *see* EX-1005, Fig. 2, the features chosen by the method depicted in Figure 1 cannot be based on the values of the static data points extracted from the software application to be classified.

67. Similarly, because Ideses only discloses feature "pruning" that has already occurred before an application is even received for classification, that pruning cannot have any logical nexus to, and thus cannot be "based on," the values of any static data points contained within, and extracted from, that application.

68. Petitioner and Dr. Tsudik argue that Ideses' disclosure that a feature may be selected based on "coefficients" that appear to serve as proxies for its "classification effectiveness" or "significance" satisfies this limitation. (Petition 30; EX-1004 at 40-41.) I disagree with this mapping because Ideses is unambiguous that these coefficients are "gathered during the classification *training*" stage. (EX-1005 [101]; *see also id.* [0097] ("the classifier training generates a vector and/or matrix of coefficients").) As such, according to Ideses, these coefficients have already been generated prior to the system's receipt of an application to be classified. Therefore, these coefficients cannot represent the "values" of the static data points that will be extracted from that application when it is classified.

69. Accordingly, it is my opinion Ideses fails to disclose feature selection at the classification stage "based on…a value of one or more static data points," as required by Claim 1.

2. **Ideses does not render claims 2-20 obvious**

70. Regarding independent claims 10 and 17, Petitioner presents no independent argument beyond those it presents for claim 1. (*See* Petition 42, 44-46.) Accordingly, Petitioner's arguments regarding independent claims 10 and 17 fail for the same rea-sons as those regarding claim 1.

71. Regarding the dependent claims, because Petitioner has failed to establish a reasonable likelihood of success on any of the independent claims, it cannot meet its burden for the challenged dependent claims.

B. Ground 2: Mori Fails To Render Claim 1 Obvious

72. As I explain above at Part IV, Claim 1 of the '844 patent is directed to feature selection at the classification rather than the training stage, and that selection, i.e. turning features "on" or "off," is performed based on the extracted content (i.e. the values of the "static data points") of the executable being evaluated. (EX-1001 7:57-8:22; 12:54-13:4.) In my opinion, Mori is inapposite to Claim 1 because, in Mori feature selection (the M features that it learns to classify during training) occurs while the classifier is being "trained" within its "supervised machine learning system," and the set of features that will be used to classify an application has already

been determined (i.e. the super-set of M readable words that were extracted from its training set of executables).

1. Mori fails to render claim obvious because it does not teach or suggest turning features on or off at the classification stage

73. In my opinion, Mori fails to render claim 1 obvious because it does not teach turning the features of a feature vector "on" or "off" at the classification stage.

74. To the contrary, in Mori no features are turned "on" or "off" at the classification stage. The M features of the feature vector extracted from the executable that are to be classified at the classification stage are the same set of M features extracted from each executable during the training stage. The classifier looks for the same set of M readable words, within the executable to be classified, that were extracted from the "training set" of executables. That occurs whether those words actually exist within the new executable or not.

75. The classifier sets the value of each feature determined at the training stage in one of two ways:1) it sets "0" or "1" to signify the presence ("1") or absence ("0") in the executable of the word corresponding to that feature (setting "0" if it is absent but "1" if it is present); or 2) it sets a value representing the frequency within the executable of the word corresponding to that feature. Thus, regardless of whether the word corresponding to each of the predefined M features exists within the executable being classified or not, Mori's classifier evaluates ***every one of the M***

features set at the training stage and designate a value for it. No feature during the training stage is turned "on" or "off" for the subsequent classification.

76. As such, in my opinion, Mori's classifier does not selectively "turn on" or "off" any features of the feature vector at the classification stage. Instead, it always extracts and evaluates the same, frozen set of M features that were defined during training.

77. Petitioner and Dr. Sudik set forth two theories for how Mori's classifier turns a feature "on" or "off": (1) the classifier sets the value of a feature to "0" when the corresponding word is absent and "1" when it is present (Petition 54-55; EX-1004 at 69-70) and (2) the classifier turns "off" a feature by not including it in the feature vector in the first place, when that word does not appear in an executable (Petition 56; EX-1004 at 71). As an initial matter, the Petitioner acknowledges (Petition 56) that these two techniques are logically equivalent because the classifier can simply interpret the absence of a value for a feature as a 0-value, a conventional technique known as "vector padding." Saliently, however, neither technique turns a feature "on or off" within the meaning of claim 1 because the set of M features that are evaluated for any given executable is the same set of features identified in the training stage; none of the training stage features are turned "off" for the classification phase.

78. First, Petitioner and Dr. Tsudik argue that Mori's classifier turns a feature "off", within the meaning of Claim 1, when it sets the value of that feature to "0." (Petition 54-55; EX-1004 at 69-70.) That is erroneous because the feature is still being evaluated at the classification stage; it is not turned off in the sense that the feature is not evaluated during classification. A feature that has a value ascribed to it, even a value of "0," is by definition a feature with a value that has been extracted during classification and whose value is considered by the classifier.

79. The feature is used by the classifier classification and the value of "0" assigned by the classifier simply indicates that the readable word corresponding to that feature does not appear in the executable. Mori's classifier considers that feature in classifying the executable no less than it considers the features whose value is not 0. In either case, Mori's classifier spends time and resources extracting a value for the feature, and the system still feeds the extracted 0-value into the classifier for use in classification. Thus, in my opinion, Mori does not turn a feature "off" by setting the feature's value to 0.

80. Second, Petitioner and Dr. Tsudik argue that Mori turns a feature "off" by excluding it from the feature vector entirely, rather than by setting its value to "0". (Petition 56; EX-1004 at 71.) Petitioner and Dr. Tsudik rely, for this argument, entirely on their assumption that the example feature vectors that are only partially depicted (each ends with an ellipsis) in Table 5 represent different sets of features

from one another, and that this implies that Mori's trained classifier can (in the classification stage) extract and classify feature vectors with different sets of features, without being retrained. (*See id.* ("One of skill in the art…would have understood that if a word is not present, the 'static data point' for that word is 'turned off' since it is not included in the generated feature vector for that file (i.e. the tf-idf value is 0 because the tf portion is 0.").).

81. That assumption is contradicted by Mori's full disclosure as understood by a POSITA. For instance, (as discussed above) Mori explicitly discloses that features with a value of "0" are *included* in the feature vector when classification occurs rather than being *excluded* from it ("off"): the value of a feature, recites Mori, can "be defined as whether a word existed or not by assigning it a '0' or '1'." (EX-1006 [0030]. Mori also discloses that the "tf-idf" value, a measure of the frequency with which a word appears within an executable, will have a value of 0 when the corresponding readable word does not appear in the executable. (*Id*.). In other words, *every* single embodiment of Mori discloses that the features in the training set are all evaluated during classification (none are turned off), and that when a word does not exist within an executable being classified, it is *included* within the feature vector, and assigned a value. Moreover, a POSITA would understand that the feature vectors shown in Table 5 do not suggest that Mori teaches a trained classifier that can operate on feature vectors with different sets of features. Machine learning classifiers are

normally trained to classify feature vectors of a single length, and Mori provides no description of a classifier that *could* operate on vectors of different sizes, let alone *how* it would do so. To the contrary, Mori expressly teaches that these feature vectors would be converted into vectors of the same length, by "assigning [the feature corresponding to a missing word] "0,"." EX-1006 at [0030]. Consequently, because Mori does not teach removing any features set at the training stage so that they are not used at the classification stage, Mori does not teach turning any features "off" within the meaning of claim 1.

82. My opinion would be no different even if the feature vectors partially depicted in Table 5 were of variable length, such that they each represented a distinct sets of features. First, Mori explicitly describes these feature vectors as having been extracted at the training stage, not the classification stage, because the right-most column of Table 5 indicates that the "class" of each executable is already known, which is consistent with these executables being part of the "training set" that is introduced into Mori's "supervised machine learning" system. (EX-1006 [0030]-[0031], [0050], [0056]-[0059].)

83. Second, even if these feature vectors had been extracted at the classification stage, a POSITA would not infer that Mori's classifier could, in a deviation from general principles of machine learning, operate on feature vectors of different lengths. In fact, a POSITA would understand that Mori's disclosure in Table 5 of

feature vectors that appear to ***omit*** values for words absent from an executable is consistent with Mori's disclosure that a ***value of "0"*** is assigned to the corresponding feature of the feature vector for a word that is absent from an executable. (EX-1006 at [0030].). That is so because, as Mori explains, variable-length feature vectors that omit 0-value features are logically equivalent to, and converted into, fixed-length vectors that include the 0-values. (EX-1006 at [0030]). This is a well-known technique called "vector padding," which would insert 0-values for each missing feature. Vector padding is necessary before a classifier, which is generally only trained to operate on feature vectors of a single length, can classify input feature vectors of variable-lengths. Therefore, in my opinion, the feature vectors shown in Table 5 teach that the same features set in training are used during classification and no features are turned "off" and not used during classification. It does not teach a POSITA that the classifier shown in Mori operates on feature vectors that have excluded the features corresponding to words that do not exist in an executable to be classified.

84. Mori simply does not teach turning features "on" or "off" for purposes of classification, and discloses no method by which it may do so. Instead, Mori's classifier evaluates all ***M*** features that were used to train that classifier during the training stage, and Table 5 can only be construed, in accordance with the general principles of machine learning and with Mori's own disclosure, as representing "padded" vectors of the same length. It is thus my opinion that Mori does not teach

turning any features "on or off," within the meaning of claim 1. Consequently, Mori cannot render obvious the claimed idea of turning features "on or off based on" any content within that executable, let alone "based on whether a value of one or more … extracted static data points [from the executable] is within a predetermined range," as recited in claim 1.

2. Mori fails to render claims 8-10 and 17 obvious

85. Regarding independent claims 10 and 17, Petitioner presents no independent argument beyond those it presents for claim 1. (*See* Petition 58-61) Accordingly, in my opinion, Petitioner's arguments regarding independent claims 10 and 17 fail for the same reasons as those regarding claim 1.

86. Regarding the dependent claims, it is my opinion that because Petitioner has failed to establish a the obviousness of any of the independent claims, it has not done so for any of the dependent claims.

C. Ground 3: The Combination Of Mori And Ideses Does Not Render Claims 1-20 Obvious

87. In my opinion, none of the 20 claims of the '844 patent are rendered obvious by the combination of Mori and Ideses. That is so for three reasons. First, Petitioner does not explain what the combination of Mori and Ideses would be; second, Petitioner does not explain why a skilled artisan would have been motivated to make that combination, and third, no combination of Ideses and Mori would render the claimed invention obvious because, as I explain above, neither reference teaches,

suggests, or renders obvious the idea of turning features on or off during the classification stage.

1. **Petitioner does not explain the proposed combination.**

88. Petitioner and Dr. Tsudik do not explain their hypothetical combination of Ideses and Mori. Petitioner uses Ideses as the primary reference. (Petition 61 (if "Ideses does not sufficiently teach or suggest any element of any of the challenged independent claims, Mori discloses these features").) And Dr. Tsudik makes the conclusory allegation that it would have been obvious to combine Ideses with "Mori's method of detecting encrypted/packed malware applications by focusing on 'readable [character] strings'." EX-1004 at ¶180. But neither Dr. Tsudik nor Petitioner explain how the combined system would operate—i.e., what it means "to have combined Mori's 'readable strings' malware analysis technique with Ideses' malware detection method." (Petition 60.)

89. In my opinion, because the Petitioner and Dr. Tsudik fail to articulate the proposed combination, they have failed to demonstrate that the combination renders any claim of the '844 patent obvious.

2. **Petitioner does not provide a sufficient motivation to combine Ideses and Mori.**

90. In my opinion, Petitioner and Dr. Tsudik have failed to articulate a sufficient motivation to combine Ideses and Mori.

91. Dr. Tsudik states that a "POSITA faced with the problem presented in Ideses (resource-limited client devices) would have been motivated to use Mori's static analysis technique using 'readable strings'" because Mori's technique can be "used without 'decryption, disassembly, or debuggers.'" EX-1004, at ¶ 186.

92. But Dr. Tsudik's statement contradicts Petitioner's earlier argument that Ideses already teaches that "feature extraction would be conducted without decrypting or unpacking the executable file, and thereby obviates a POSITA's need to seek out Mori's decryption-less technique. (Petition 21 (emphasis added). Similarly, Dr. Tsudik states that, in Ideses, "features are extracted 'without decrypting or unpacking the executable file.'" EX-1004 at ¶ 70 (emphasis added).)

93. In my opinion, Dr. Tsudik position that Ideses already teaches extracting features "without decrypting or unpacking the executable files" (*id.* ¶ 70) is irreconcilable with his later position that a skilled artisan would have been motivated to combine Ideses and Mori because Mori's technique can be "used without decryption, disassembly or debuggers," because Dr. Tsudik alleges that the benefits he identifies in Mori were already realized in Ideses. As such, it is my opinion that neither Dr. Tsudik nor the Petitioner have articulated a sufficient motivation to combine the two references.

3. No combination of Ideses and Mori can render the independent claims obvious because neither reference teaches turning features on or off during the classification stage.

94. Finally, as I have explained above with respect to Ideses and Mori separately, neither reference teaches, suggests, or renders obvious the concept of turning features "on or off based on … a value of … one or more static data points" extracted from the executable being classified at the classification stage. Neither Dr. Tsudik or the Petitioner identify any additional disclosures in the combination of Ideses and Mori that would overcome their individual deficiencies.Thus, in my opinion, any hypothetical combination of Ideses and Mori would not render any of the claims of the '844 patent obvious.

VII. CLAIMS AND/OR LIMITATIONS NOT ADDRESSED IN THIS DECLARATION

95. It is my understanding that this declaration is being provided in support of Patent Owner's Preliminary Response. I have provided analysis of key limitations not included in the prior art. The fact that I have not analyzed particular limitations or claims in no way indicates or supports that these limitations or claims are disclosed or rendered obvious by the prior art, and I reserve the right to supplement my analysis if asked to do so.

VIII. SECONDARY OBJECTIVE INDICIA OF NONOBVIOUSNESS

96. I understand that in determining obviousness of the claimed invention, objective evidence of secondary considerations pertaining to nonobviousness must

be considered, including longfelt but unsolved needs, failure of others, commercial success, etc. (*In re Huang*, 100 F.3d 135, 139 (Fed. Cir. 1996) (citing *Graham v. John Deere Co.*, 383 U.S. 1, 1718 (1966).) I further understand that these indicia constitute independent evidence of nonobviousness that can be the most probative evidence in that regard "in avoiding the trap of hindsight when reviewing, what otherwise seems like, a combination of known elements." (*Leo Pharm. Prods., Ltd. v. Rea*, 726 F.3d 1346, 1358 (Fed. Cir. 2013).)

97. As demonstrated by my summary of the prior art, above, there was at least a longfelt and persistent need prior to the '844 patent to improve upon prior art classification performance by calibrating feature vectors, at the classification stage, based on the content of the executable to be classified. (See Section III, supra.) Prior attempts to develop such methods were unable to perform such dynamic calibration at the classification stage. (*See* Section III, supra.) Where others failed, the '844 Patent succeeded. (*See* Section IV(A), supra.)

98. Thus, these secondary indicia only further support my opinion that the '844 Patent was nonobvious.

IX. CONCLUSION

99. For at least the foregoing reasons, it is my opinion that Petitioner's grounds do not render obvious any of the challenged claims of the '844 Patent.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Executed this 16th day of May, 2023 in San Francisco, CA.

Respectfully submitted,



Sam Malek, Ph.D.