

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

SOPHOS LTD. and SOPHOS INC.,
Petitioner,

v.

OPEN TEXT INC.,
Patent Owner.

PGR2023-00031
Patent 11,409,869 B2

Before JUSTIN T. ARBES, BARBARA A. PARVIS, and
GARTH D. BAER, *Administrative Patent Judges*.

PARVIS, *Administrative Patent Judge*.

DECISION
Denying Institution of Post-Grant Review
35 U.S.C. § 324

I. INTRODUCTION

Sophos Ltd. and Sophos Inc. (collectively “Petitioner”) filed a Petition (Paper 1 (“Pet.”)) requesting post-grant review of claims 1–59 (the “challenged claims”) of U.S. Patent No. 11,409,869 B2 (Ex. 1001, “the ’869 patent”), along with the supporting Declaration of Gene Tsudik, Ph.D. (Ex. 1004). Open Text Inc. (“Patent Owner”) filed a Preliminary Response Paper 6 (“Prelim. Resp.”), along with the supporting Declaration of Sam Malek, Ph.D. (Ex. 2001).

We have authority to determine whether to institute a post-grant review under 35 U.S.C. § 324 and 37 C.F.R. § 42.4. A post-grant review may not be instituted “unless . . . the information presented in the petition . . . , if such information is not rebutted, would demonstrate that it is more likely than not that at least 1 of the claims challenged in the petition is unpatentable.” 35 U.S.C. § 324 (2018).

Having considered the arguments and evidence presented by Petitioner and Patent Owner, we determine that Petitioner has not demonstrated that it is more likely than not that at least one of the claims challenged in the patent is unpatentable. Accordingly, we do not institute a post-grant review of the challenged claims of the ’869 patent.

II. BACKGROUND

A. *Real Parties-in-Interest*

Petitioner identifies as the real parties-in-interest the following: Sophos Ltd., and Sophos Inc. Pet. 2. Patent Owner states the “real parties-in-interest are Open Text Inc. (‘Patent Owner’) and Webroot LLC.” Paper 4, 1¹

¹ Patent Owner identifies as related matters cases pending in the Western District of Texas naming Webroot, Inc. as the Plaintiff. Paper 4, 1. If Patent

Patent Owner also states “[o]ut of an abundance of caution, Patent Owner further identifies the following affiliates: Carbonite, LLC; Open Text Corporation; and Open Text Holdings, Inc.” *Id.*

B. Related Matters

Each party identifies a judicial matter that would affect, or be affected by, a decision in this proceeding. In particular, the parties inform us that the ’869 patent is asserted in the following district court proceedings: *Webroot Inc, et al. v. Sophos Ltd.*, No. 6:22-cv-240 (W.D. Tex.); *Webroot, Inc. et al. v. Trend Micro, Inc.*, No. 6:22-cv-239; *Webroot, Inc. et al. v. CrowdStrike Holdings, Inc. et al.*, No. 6:22-cv-241; *Webroot, Inc. et al. v. AO Kaspersky Lab*, No. 6:22-cv-243; and *Webroot, Inc. et al. v. Forcepoint LLC*, 6:22-cv-00342. Pet. 3; Paper 4, 1. The parties also identify the following PTAB proceedings concerning the ’869 patent’s parent patent, U.S. Patent No. 10,599,844: IPR2023-00528 and IPR2023-00662. Pet. 3; Paper 4, 2. We also note that three other petitions have been filed challenging the ’869 patent in IPR2023-01053, IPR2023-01160, and IPR2023-01380.

C. The ’869 Patent

The ’869 patent is titled “Automatic Threat Detection of Executable Files Based on Static Data Analysis.” Ex. 1001, code (54). The ’869 patent issued from U.S. Application No. 16/791,649, filed on February 14, 2020 (“the ’649 application”). Ex. 1001, codes (21), (22). The ’649 application was filed as a continuation of U.S. Application No. 14/709,875, filed on

Owner intended to identify Webroot, Inc. rather than Webroot LLC, Patent Owner should file corrected mandatory notices.

May 12, 2015, which is now U.S. Patent No. 10,599,844 B2 (“the ’844 patent”). *Id.*, code (63).²

The ’869 patent describes extracting “static data points” from the file and applying to them a classifier that was trained from known malicious, benign, and unwanted executable files. *Id.* at 1:34–40. The general architecture of the system is shown in Figure 1, reproduced below:

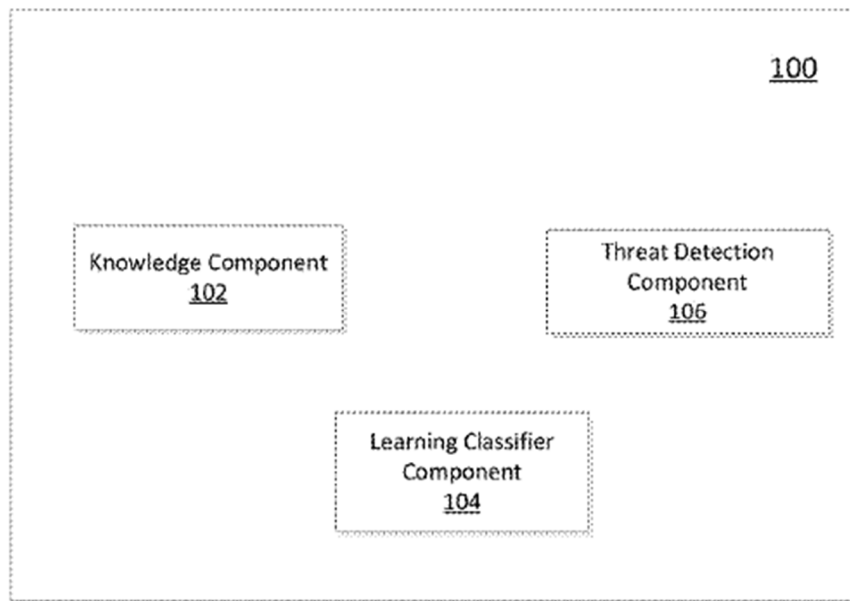


FIG. 1 illustrates exemplary system 100 showing components for threat detection. Id. at 1:59–61.

System 100 comprises knowledge component 102, learning classifier component 104, and threat determination component 106. *Id.* at 3:31–34. “To classify executable files . . . the knowledge component 102 collects static data on a large variety of executable files.” *Id.* at 4:29–31. The data

² Because the earliest possible effective filing date for the ’869 patent is after March 16, 2013 (the effective date for the first inventor to file provisions of the America Invents Act) and the Petition was filed not later than 9 months after its issue date, the ’869 patent is eligible for post-grant review. *See* 35 U.S.C. § 321(c).

collected “yields a knowledgebase that may be used to periodically train a classifier, e.g. learning classifier component 104,” which “may be used to classify an executable file as one of the following classifications: malicious, benign or potentially unwanted.” *Id.* at 4:39–46. “Continuous update of the data maintained by the knowledge component 102 in conjunction with on-going re-learning/re-training of the learning classifier component 104 ensures that system 100 is up-to-date on current threats.” *Id.* at 4:63–67.

The collected data “is analyzed to identify static data points that may indicate one of a malicious file, a benign file or a potentially unwanted file.” Ex. 1001, 5:8–11. Static data points “include, but are not limited to: header information, section information, import and export information, certificate information, resource information, string and flag information, legal information, comments, and/or program information (e.g., APIs and DLLs), among other examples.” *Id.* at 5:62–67. Table 1.1 of the ’869 patent, showing possible static data points, is reproduced below:

TABLE 1.1

Numeric values	Nominal values	Strings/Byte sequences	Boolean values
File size	initialize	Comments	Address Of Entry Point Anomaly
linker version	Un-initialize	company name	Image Base Anomaly
code size	entry point	file description	Section Alignment Anomaly
OS version	subsystem	internal name	Size Of Code Mismatch Anomaly
image version	file subtype	legal copyright	Low Import Count Anomaly
subsystem	language	original file	Entry Point Anomaly
version			
file version	file flags masks	private build	certificate Validity
number			
product version	file flags	product name	Certificate Exception
number			
size of heap	file OS	special build	Code Characteristics Anomaly
size of stack	file type	product version	Code Name Anomaly
size of image	machine type	file version	Count Anomaly
PE header time	PE type	package code	Data Characteristics Anomaly
Section Entropy	section counts	product code	Data Name Anomaly
Sections count	DLL count	export DLL name	Export Exception
	DLL functions	assembly version	Large Number of DLLs Anomaly
	data directory	Certificate Issuer	flag DLL Name Anomaly
	export count	Certificate	Number of Functions
		Subject	Anomaly
	Earliest Data Byte	Imports	Function Name Anomaly
	resources	Exports	PE Header Anomaly
	resources language	Section Names	High Section Count Anomaly
	resource Encoding	Non-resource	PE Magic Validity
		section strings	
	resource code		Resource Exception
	page		
	resource size		VR Code Ratio Anomaly
	DLL		Import Exception
	characteristics		

Table 1.1 shows possible static data points, organized by category.

The learning classifier component 104 “interfaces with the knowledge component 102 and a threat detection component 106 for the system 100 to evaluate an executable file as a possible threat,” such as by applying “programming operations or machine learning processing to evaluate static data points extracted from a file to be analyzed.” Ex. 1001, 7:28–34. It “builds a learning classifier based on information from the knowledge component 102 including the extracted static data points of a file and the information used to train/re-train the learning classifier.” *Id.* at 7:34–42.

“[T]he learning classifier can adaptively set features of a learning

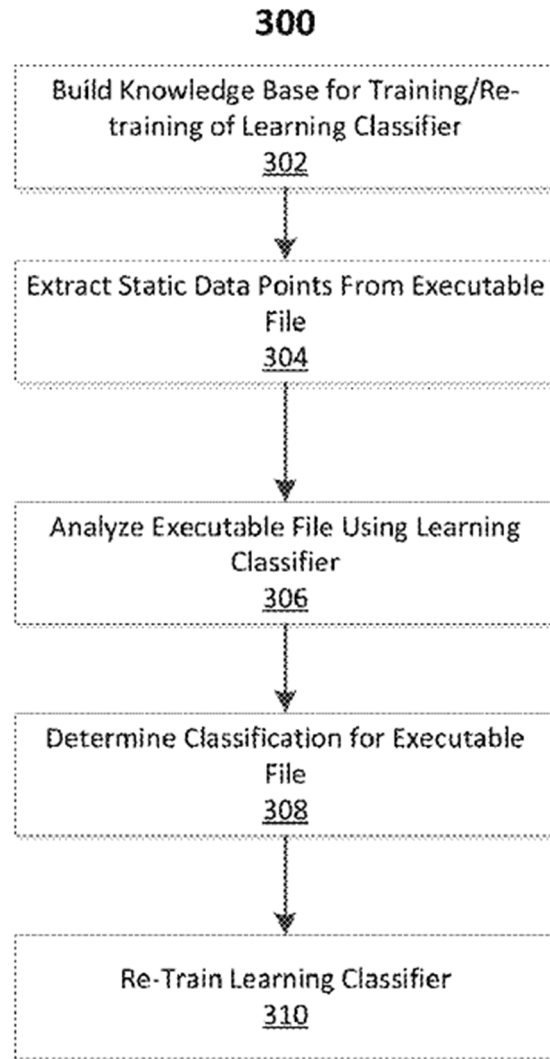
classifier based on the static data points extracted for a file.” Ex. 1001, 7:43–45. For example, “ranges of data can be identified for static data points that may enable the learning classifier to controllably select (e.g., turn on/off) features for evaluation by learning classifier.” *Id.* at 7:45–48. As a more specific example, “where a static data point extracted for evaluation is a file size . . . , the learning classifier may turn on/off features for evaluation based on whether . . . the file size . . . is within a certain range.” *Id.* at 7:48–52.

“The learning classifier component 104 processes and encodes collected/extracted data from a file to make the static data points suitable for processing operations” and “the different encoded data may be combined into one or more final feature vectors.” *Id.* at 8:8–10, 8:31–34.

The learning classifier component 104 then “provides the one or more feature vectors as input to a support vector machine (SVM),” which “may then perform data analysis and pattern recognition on the one or more feature vectors” and “build a probabilistic model that indicates whether or not a file may be malicious.” *Id.* at 9:27–34.

The threat detection component 106 “evaluates results of the processing performed by the learning classifier component 104 to make a determination as to whether the executable file is malicious, benign or a potentially unwanted file/application.” *Id.* at 9:55–59. For example, “a final score (e.g., probability value/evaluation) [may be] determined based on feature vector processing performed by the learning classifier component 104 and compared with a probability threshold value for determining whether an executable file is malicious.” *Id.* at 10:3–28.

Figure 3, reproduced below, is a flowchart showing exemplary threat detection method 300:



“FIG. 3 illustrates an exemplary method 300 for implementation of threat detection systems and methods.” Ex. 1001, 1:65–67.

The method begins at “operation 302 where a knowledge base is built for training/retraining of a learning classifier used to detect threats in executable files . . . from data collected and evaluated related to known malicious executable files, known benign executable files and potentially unwanted executable files” by knowledge component 102. *Id.* at 11:60–67. The knowledge base “may be used to automatically train/re-train one or more learning classifiers used to evaluate threats in executable files based on

the known malicious executable files, known benign executable files and the potentially unwanted executable files.” *Id.* at 11:67–12:4.

“When an executable file is identified for evaluation,” the method proceeds to operation 304, “where a one or more static data points are extracted from [the] executable file” and the file “is analyzed using machine learning processing as described with respect to the knowledge component 102 . . . to collect/extract static data points . . . for evaluation.” Ex. 1001, 12:11–17. This may happen with or without decrypting or unpacking the file. *Id.* at 12:17–19.

At operation 306, “the executable file is analyzed for threats” by “applying a learning classifier (e.g., the learning classifier generated during performance of operation 302) to the plurality of static data points extracted from the file.” Ex. 1001, 12:43–51. The system may “generat[e] at least one feature vector from the plurality of static data points extracted using the learning classifier trained by the knowledge base” and the “data may be parsed and encoded for machine learning processing.” *Id.* at 12:55–60.

As noted above, “generation of the feature vector may comprise selectively setting features of the learning classifier based on the one or more of [the] static data points extracted” and “[f]eatures of the generated feature vector may be weighted based on classified categories identified by the knowledge base . . . and the plurality of static data points extracted from the file.” *Id.* at 12:61–67. In addition, “one or more features of the feature vector may be selectively turned on or off based on evaluation of whether a value of a static data point is within a predetermined range.” *Id.* at 12:67–13:3. The feature vector is evaluated “using linear or nonlinear support vector processing to determine a classification for the executable file, for example whether the executable file is harmful, benign, or unwanted.” *Id.* at

13:7–11.

At operation 308, “a determination is made as to a classification of the executable file,” i.e., “whether the executable file is harmful (e.g., malicious, malware) or not based on results of the analysis of the executable file (e.g., using machine learning processing by a learning classifier).” Ex. 1001, 13:12–18. The “results of the analysis of [the] executable file may be data obtained from learning classifier, such as, for example an SVM, processing data.” *Id.* at 13:18–20. The classification “may be based on any type of analytical, statistical or graphical analysis, or machine learning processing.” *Id.* at 13:32–35.

“At any point in time, operation 310 may occur where a learning classifier used for threat detection processing is re-trained” to “ensure that the threat detection application/service is up to date and able to accurately detect new threats.” *Id.* at 13:40–42.

D. Illustrative Claim

Petitioner challenges claims 1–59 of the ’869 patent. Pet. 1. Claims 1, 10, 15, 18, 32, and 46 are independent. Independent claim 1, reproduced below,³ is illustrative of the claimed subject matter.

- [1.pre] A computer-implemented method comprising:
 - [1.a] identifying, by a knowledge module, static data points that may be indicative of either a harmful or benign executable file;
 - [1.b] associating, by the knowledge module, the identified static data points with one of a plurality of categories of files, the plurality of categories of files including harmful files and benign files;
 - [1.c] identifying an executable file to be evaluated;

³ We use Petitioner’s annotations. Pet. 16–36.

- [1.d] extracting, by the knowledge module, a plurality of static data points from the identified executable file;
- [1.e] generating a feature vector from the plurality of static data points using a classifier trained to classify the static data points based on training data, the training data comprising files known to fit into one of the plurality of categories of files, wherein one or more features of the feature vector are selectively turned on or off based at least in part on evaluation of whether a value of one of the plurality of static data points is within a predetermined range; and
- [1.f] providing the generated feature vector to one or more support vector machines to build a probabilistic model that indicates whether the executable file fits into one of the categories of files.

Ex. 1001, 16:10–33.

E. Evidence

Petitioner relies on the patent document references summarized in the table below.

Name	Patent Document	Exhibit
Ideses	US 2015/0213376 A1, filed Jan. 29, 2015, published July 30, 2015	1005
Mori	JP 2012-27710 A, filed July 23, 2010, published Feb. 9, 2012	1006

F. Asserted Grounds

Petitioner asserts that the challenged claims of the '869 patent are unpatentable based on the following grounds summarized in the table below:

Claims Challenged	35 U.S.C. §⁴	Reference(s)/Basis
1–59	103	Ideses

⁴ Because the challenged claims of the '869 patent have an apparent effective filing date after March 16, 2013, the 35 U.S.C. § 103 provision of

Claims Challenged	35 U.S.C. § ⁴	Reference(s)/Basis
1–6, 8–24, 26, 27, 29, 30, 32–38, 40, 41, 43, 44, 46–52, 54, 55, 57, 58	103	Mori
1–59	103	Ideses, Mori
1–59	112(b)	Indefiniteness
8, 14, 17, 29, 43, 57	112(b)	Indefiniteness
8, 14, 17, 29, 43, 57	112(a)	Written description

III. ANALYSIS

A. Legal Standards

A patent claim is unpatentable under 35 U.S.C. § 103 if the differences between the claimed subject matter and the prior art are such that the subject matter, as a whole, would have been obvious at the time of the effective filing date of the claimed invention to a person having ordinary skill in the art to which said subject matter pertains. *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). The question of obviousness is resolved on the basis of underlying factual determinations including: (1) the scope and content of the prior art; (2) any differences between the claimed subject matter and the prior art; (3) the level of ordinary skill in the art; and (4) when in evidence, objective evidence of nonobviousness, i.e., secondary considerations. *Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966).

B. Level of Ordinary Skill in the Art

Petitioner asserts the following:

A [person of ordinary skill in the art (POSITA)] on May 12, 2015, would have [had] a Bachelor’s degree in Computer Science, Computer Engineering, Electrical Engineering, or an

the Leahy-Smith America Invents Act (“AIA”), Pub. L. No. 112-29, §§ 3(b)–3(c), 3(n)(1), 125 Stat. 284, 285–87, 293 (2011) applies.

equivalent discipline, and at least one year's worth of experience relating to static data analysis and malware detection; alternatively, 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.

Pet. 10 (citing Ex. 1004 ¶ 16). Patent Owner does not dispute Petitioner's proposal. Prelim. Resp. 33.

Considering the subject matter of the '869 patent, the background technical field, and the asserted prior art, we agree with Petitioner's proposed qualifications for an ordinary level of skill, except we decline to adopt "at least" as that language is vague and open-ended. Otherwise, Petitioner's proposed definition is consistent with the level of skill reflected in the Specification of the '869 patent and the asserted prior art references. *See Okajima v. Bourdeau*, 261 F.3d 1350, 1355 (Fed. Cir. 2001).

Accordingly, for purposes of this Decision, and based on the current record, except as noted above, we adopt Petitioner's assessment of the level of skill for one of ordinary skill in the art.

C. Claim Construction

We apply the same claim construction standard used in district court actions under 35 U.S.C. § 282(b), namely that articulated in *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (en banc). *See* 37 C.F.R. § 42.100(b). In applying that standard, claim terms generally are given their ordinary and customary meaning as would have been understood by a person of ordinary skill in the art at the time of the invention and in the context of the entire patent disclosure. *Phillips*, 415 F.3d at 1312–13.

Petitioner states "the parties stipulated, while reserving appellate rights, that the District Court's claim construction for the '844 Patent would apply to the '869 Patent." Pet. 8–9 (citing Ex. 1010). Petitioner also states

that it maintains “that the terms relating to whether ‘character strings’ are ‘within a predetermined range’ are indefinite and lack written description,” although “the District Court did not find the term indefinite but has not yet issued a full claim construction order.” *Id.* at 8 (Ex. 1007, 2, 18).

Patent Owner states the “parties agree that no claim terms need be construed for this petition.” Prelim. Resp. 31. Patent Owner also states in “analyzing the ’869 patent,” the “district court held that ‘static data points’ is not indefinite, and that the phrase takes its plain and ordinary meaning.” *Id.* (citing Ex. 2003, 19). Patent Owner also states the “district court held that the phrase ‘within a predetermined range’ carries its plain and ordinary meaning, and it defined ‘range’ as a ‘numerical range or a subset of a “particular scale” with at least a lower limit or an upper limit, but not necessarily both.’” *Id.* (citing Ex. 2003, 19).

We address the parties’ contentions regarding Petitioner’s indefiniteness grounds below. We determine no other determinations are needed regarding claim construction to resolve whether to institute post grant review. *See Realtime Data, LLC v. Iancu*, 912 F.3d 1368, 1375 (Fed. Cir. 2019) (“The Board is required to construe ‘only those terms . . . that are in controversy, and only to the extent necessary to resolve the controversy.’” (quoting *Vivid Techs., Inc. v. Am. Sci. & Eng’g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999))).

D. Unpatentability of Claims 1–59 under 35 U.S.C. § 103 as Obvious

Petitioner asserts that claims 1–59 are unpatentable as obvious over Ideses. Pet. 15–59. Petitioner also asserts that claims 1–6, 8–24, 26, 27, 29, 30, 32–38, 40, 41, 43, 44, 46–52, 54, 55, 57, and 58 are unpatentable as obvious over Mori. Pet. 59–82. Petitioner further asserts claims 1–59 are

unpatentable as obvious over the combination of Ideses and Mori. Pet. 83–86. Patent Owner disputes Petitioner’s assertions. *See generally* Prelim. Resp.

We begin with an overview of Ideses and Mori. We then turn to the parties’ contentions.

1. Ideses

Ideses is titled “Methods and Systems for Generating Classifiers for Software Applications.” Ex. 1005, code (54). Ideses “relates to . . . methods and systems for generating classifiers for software applications and, more specifically . . . to methods and systems for generating classifiers for software applications based on large feature vectors.” *Id.* ¶ 2.

Figure 1 of Ideses is reproduced below.

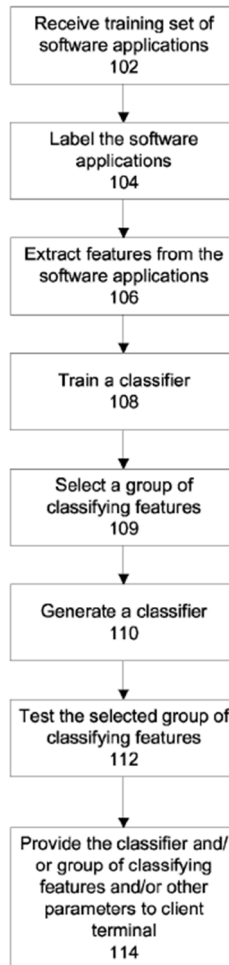


Figure 1 is a “flowchart of a method of generating a pruned feature list for classification.” Id. ¶ 39.

At step 102, the system receives multiple software applications for training a classifier and, at step 104, the applications may be “labeled with a classification type based on desired software application classification categories.” *See id.* ¶¶ 80–81.

At step 106, features may be extracted from the software applications by a feature extractor module and “[o]ptionally, a feature vector [may be] extracted.” *See id.* ¶ 86. The extracted features may be any feature that describes the software application, including static meta data, information regarding the executable code, or behavioral features. *See id.* ¶ 89. The

features may be stored as feature vectors, which may be used for training a classifier. *See id.* ¶ 91.

At step 108, “a classifier [may be] trained based on the set of extracted features (block 106) and software classification labeling (block 104).” *Id.* ¶ 94. At step 109, “a group of classifying features is selected, for example, by a combination and/or selection module 306.” *See id.* ¶ 98. “[E]xtracted features [may be] identified for pruning to generate the group of classifying features,” and “the selection [may be] based on identifying a classification effectiveness rank for each of the features of the training software applications” or “based on identifying resource requirements for each of the features of the training software applications.” *See id.* ¶¶ 98, 99.

The selection module may use “the parameters and/or coefficients gathered during the classification training” and “the group of classifying features [may be] selected based on the significance of the extracted features,” such that “[s]ignificant features, such as those that have a large effect on the classification outcomes, may be retained” and “[i]nsignificant features, such as those that have a small, negligible or no effect on classification outcome may be pruned.” Ex. 1005 ¶¶ 101, 106.

“[F]eatures (or their equivalents) that do not contribute (or do not significantly contribute) to the classification process [may be] pruned or not included,” “features that correspond to coefficients having a value of zero [may be] pruned or not included, for example, when classification is performed based on a support vector machine (SVM),” and “features with a low magnitude value of the coefficient [may be] identified for pruning and/or are not included.” Ex. 1005 ¶ 107.

At step 110, “a classifier is generated, for example, by a classifier generating module” that may be “based on the group of classifying

features.” Ex. 1005 ¶¶ 115–116. The feature set used to generate the trained classifier may be pruned. *See id.* ¶¶ 116–117.

Step 112 is an optional step of evaluating the “classifier performance based on the pruned feature set [as] compared to classifier performance based on the complete feature set” and “[t]he pruning may be selectively adjusted based on the testing.” *See* Ex. 1005 ¶¶ 118–119.

Finally, at step 114, “the group of classifying features, the pruned feature vector, selected coefficients, and/or trained classifier are provided to [a] mobile device” “for run-time classification of software applications by the mobile device.” *See* Ex. 1005 ¶¶ 120–122.

2. *Mori*

Mori is titled “Software Detection Methods[,] Devices[,] and Programs.” Ex. 1006, code (54). *Mori* relates to “a software detection method, apparatus and program for detecting malware or communications transmitting and/or receiving malware using supervised machine learning systems.” *Id.* ¶ 1. The method “focuses on readable strings contained in software files, collects information (characteristic vectors) that characterizes malware, such as messages contained in programs that constitute malware, host names, execution command names, function names and library names used for communication,” and “compares this content with strings contained in files being inspected to detect malware, allowing for the timely detection of malware and communications that include malware at a high rate.” *Id.* ¶ 23.

Mori describes the process as follows: “[f]irst, the file or communication data to be sent or received,” called the “Object,” is “recorded by the Object recording unit”; next, “a word extraction unit . . . extracts the printed and readable byte strings as words from the Object . . .

and uses it as a word set”; then, “by configuring a feature vector for each Object based on the . . . word set, and applying a machine learning system to the . . . feature vector, the determination function determines whether the Object is malware based on newly observed feature vectors.” Ex. 1006 ¶¶ 27–30. “[T]he feature vector can optionally be defined as whether a word existed or not by assigning it ‘0,’ or ‘1,’ the frequency per word, the normalized frequency per word, or the tf-idf value per word.” *Id.* ¶ 30.

Mori’s system is depicted in its Figure 1, reproduced below:

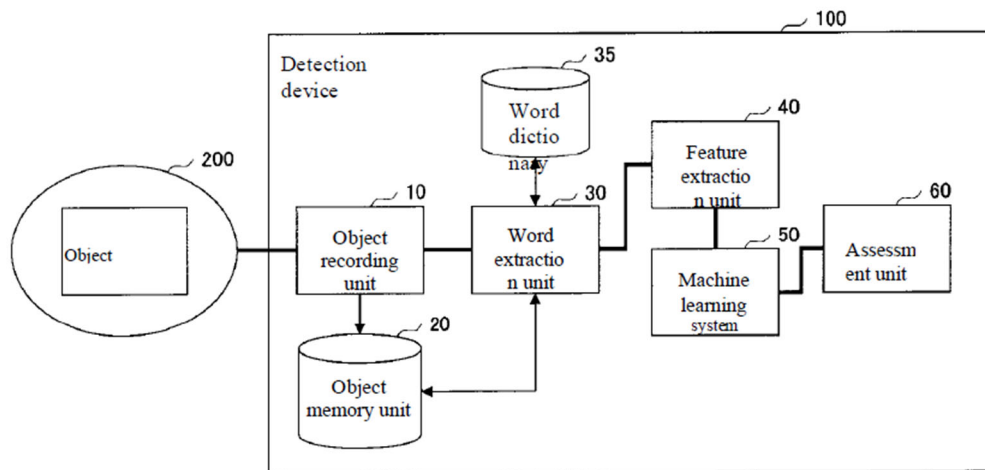


Figure 1 is an “example of a system configuration.” *Id.* ¶ 24.

As shown above, “[d]etection device 100 is connected to network 200 and consists of Object recording unit 10, Object storage unit 20, word extraction unit 30, word dictionary 35, feature extract unit 40, machine learning system 50, and assessment unit[]60.” Ex. 1006 ¶ 37.

“Word extraction unit 30 retrieves the Object from Object storage unit 20, analyzes it, and extracts a string of consecutive characters with an initial printable character code of X characters or more,” where “printable” refers, for example, “to the character codes of blank and graphic characters from 0×20 to 0×7e, excluding control characters, in ASCII codes.” Ex. 1006 ¶ 39. The strings are processed to produce word candidates such as “text,”

“suv_m,” and “set.” *See id.* ¶¶ 41–49. Then, “[w]ord extract unit 30 uses word dictionary 35 prepared in advance for the word candidates . . . , extracting only those words that match the words in word dictionary 35.” *Id.* ¶ 50.

Next, “[f]eature extraction unit 40 reads out the word set . . . for each file from the memory of word extract unit 30 and configures the feature vector.” Ex. 1006 ¶ 52. For each file X_i ($i=1, 2, \dots, N$), the feature vector is $V_i = \{v_{i1}, v_{i2}, \dots, v_{ij}, \dots, v_{iM}\}$, and, for each word w_j ($j=1, 2, \dots, M$), $v_{ij} = g(w_j, X_i, X, a)$. *Id.* ¶ 52. “X” is “the set of all files” and “a” is “an optionally definable weight parameter.” *Id.*

Finally, “[a]ssessment unit 60 applies machine learning system 50 to the feature vector V_i ” to classify the file as “malware” or a “normal file.” Ex. 1006 ¶¶ 54–55. “Specifically, each ‘malware’ and ‘normal file’ file feature vector is learned in advance, and the newly obtained file is determined to be ‘malware’ or ‘normal file’ based on that feature vector.” *Id.* ¶ 55. Mori states that “supervised learning” is “a highly accurate method” for estimating the class for a newly observed vector, “with a variety of known methods widely used, including support vector machines (SVMs), naïve Bayesian methods, neural networks, and decision trees.” *Id.* ¶ 57.

3. Analysis

The parties dispute whether Ideses and Mori teach or suggest turning features on or off at the classification stage based on the executable being classified at that stage. *See* Pet. 15–86; Prelim. Resp. 37–41 (Ideses), 44–50 (Mori). The dispute, more specifically, pertains to claim element 1.e, which recites “generating a feature vector from the plurality of static data points,” “wherein one or more *features of the feature vector are selectively turned on or off* based . . . on evaluation of whether a value of one of the plurality of

static data points is within a predetermined range” (emphasis added). Independent claims 10, 15, 18, 32, and 46 have the same recitation.

For “generating a feature vector from the plurality of static data points,” Petitioner relies on Ideses’s teachings relating to Figure 2, which “is a flowchart of a method of run-time execution of a classifier based on a pruned feature list.” Pet. 28–30 (citing, e.g., Ex. 1005 ¶¶ 123, 132, 134, Fig. 2). For “wherein one or more features of the feature vector are selectively turned on or off based . . . on evaluation of whether a value of one of the plurality of static data points is within a predetermined range,” Petitioner relies on Ideses’s teachings relating to Figure 1. *Id.* at 33–35 (citing, e.g., Ex. 1005 ¶¶ 71, 104–107, 109–110, Fig. 1). Ideses’s Figure 1 depicts a flowchart in which a classifier is *generated* at step 110 and the classifier is provided to a client terminal at step 114. Ex. 1005, Fig. 1.

Patent Owner argues that in Ideses, “at the training stage, the system selects the features that the classifier will be trained to consider” and, thus “at the later classification stage, the set of features that must be extracted and considered has already been predetermined, such that there is *no feature selection at the classification stage*.” Prelim. Resp. 37. Patent Owner argues in contrast, “the idea claimed in the ’869 patent is to turn features on or off at the *classification* stage, based on the executable being classified at *that* stage.” *Id.*

Patent Owner argues that “language of the independent claims clearly requires turning features on or off at the *classification stage* because the generated feature vector—whose features are ‘turned on or off’—is not generated until that stage.” *Id.* at 38. According to Patent Owner, “[b]ecause the recited turning of features ‘on or off’ is performed on the feature vector

at the classification stage, that turning on or off necessarily happens at the classification stage too.” *Id.*

Patent Owner argues that other claim language supports its position including that “the feature vector whose features are being turned on or off is generated ‘using a classifier [that has already been] *trained*’—past tense.” *Id.* Patent Owner also argues that its position is supported by an additional feature recited in independent claims 18, 32, and 46, specifically “that the feature vector is used to classify the identified executable.” *Id.* Patent Owner also argues that further support is found in the recitation of training and classification steps separately in claim 20. *Id.* at 38–39.

Patent Owner asserts that “[t]he benefit of turning features on or off at the classification stage is that classifier performance can be optimized *based on the executable being classified*” and “the classifier is not locked into evaluating the full set of features that was predetermined at the training stage without regard to the executable being classified at the classification stage.” Prelim. Resp. 39.

We agree with Patent Owner that the claims are directed to the classification stage, as they require extracting “a plurality of static data points” from an executable file and “generating a feature vector from [that] plurality of static data points *using a classifier*” (emphasis added). The claims also recite that the classifier has been “*trained to classify*” (emphasis added). Further, the claims recite that “features of the feature vector” are selectively turned on or off, indicating that the feature vector has been generated.

We also agree with Patent Owner that Petitioner has not shown that Ideses teaches or suggests a classification method in which one or more features of a feature vector generated from a plurality of static data points

are selectively turned on or off based on the value of the static data point. As explained in the summary of Ideses above, Ideses describes “methods and systems for generating classifiers” (Ex. 1005 ¶ 2) in which “extracted features [may be] identified for pruning to generate the group of classifying features,” and “the selection [may be] based on identifying a classification effectiveness rank for each of the features of the training software applications” or “based on identifying resource requirements for each of the features of the training software applications.” *See id.* ¶¶ 98, 99. “[T]he group of classifying features [may be] selected based on the significance of the extracted features,” such that “[s]ignificant features, such as those that have a large effect on the classification outcomes, may be retained” and “[i]nsignificant features, such as those that have a small, negligible or no effect on classification outcome may be pruned.” *Id.* ¶¶ 101, 106.

Petitioner argues that “Ideses explains that in the method depicted in Figure 1, ‘[t]he full feature list is optionally selectively pruned,’ i.e., ‘selectively turned off’ as required by [1.e],” and “[f]urther, ‘each feature of the group is selected from-the-ground-up; added to the group as opposed to removed from or retained within a larger group,’ i.e., ‘selectively turned on’ as required by [1.e].” Pet. 33 (citing Ex. 1005 ¶ 71). Petitioner also argues element 1.e is taught by Ideses’s “approaches to pruning insignificant features,” which Petitioner argues are “to selectively includ[e]/exclud[e] features based on whether a feature coefficient ‘is within a predetermined range’ as required” by element 1.e. *Id.* at 34 (citing, e.g., Ex. 1005 ¶¶ 107, 109–110).

But all of the selection and pruning described in Ideses concerns the creation of the classifier, where the features that are selected or pruned are either included or omitted from the classifier. Petitioner fails to show how

this teaches or suggests “generating a feature vector from [a] plurality of static data points using a classifier” where features are “selectively turned on or off” based on a value of one of the static data points. Even if Ideses’s omission of insignificant features could be “turn[ing] [them] on or off,” that happens in the creation of the classifier. Ideses does not teach or suggest “wherein one or more *features of the feature vector* are selectively turned on or off” subsequent to “generating a *feature vector . . . using a classifier*,” as recited in each challenged claim.

For element 1.e, Petitioner relies on Mori’s process that includes use of a classifier to detect malware, in which a feature vector is derived from the object being analyzed. Pet. 65–69 (citing Ex. 1006 ¶¶ 29–32, 49, 50, 53, 56, 57, 59, 60). However, we conclude that Petitioner has not shown how Mori teaches or suggests “wherein one or more features of the feature vector are selectively turned on or off based . . . on evaluation of whether a value of one of the plurality of static data points is within a predetermined range.”

As described in the summary of Mori above, Mori describes the process as follows: “[f]irst, the file or communication data to be sent or received,” called the “Object,” is “recorded by the Object recording unit”; next, “a word extraction unit . . . extracts the printed and readable byte strings as words from the Object . . . and uses it as a word set”; then, “by configuring a feature vector for each Object based on the . . . word set, and applying a machine learning system to the . . . feature vector, the determination function determines whether the Object is malware based on newly observed feature vectors.” Ex. 1006 ¶¶ 27–30. “[T]he feature vector can optionally be defined as whether a word existed or not by assigning it ‘0,’ or ‘1,’ the frequency per word, the normalized frequency per word, or the tf-idf value per word.” *Id.* ¶ 30.

Petitioner argues that Mori's explanation "that the feature vector can be defined 'as whether a word existed or not by assigning it '0' or '1'" means that "features are selectively turned on or off based on whether or not a word in the predetermined dictionary is present in the Object or not." Pet. 67. We do not agree because we see nothing in Mori to suggest that a feature in the vector assigned a value of 0 is not used in the evaluation. Petitioner does not adequately explain how assigning a feature a value of 0 is the same as turning it "off."

Petitioner also argues that Mori describes how "words that are present in each file are represented as '{word number: tf-idf value}' in the feature vector" and 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)." Pet. 69. We also disagree with that analysis. The claims require that "wherein one or more features of the feature vector are selectively turned on or off based . . . on evaluation of whether a value of one of the plurality of static data points is within a predetermined range." As quoted by Petitioner, Mori explains that "*words that are present in each file* are represented . . . in the feature vector," which we understand to also mean that words *not* present in the file are *not* represented in the feature vector. *Id.* (emphasis added). Thus, for words that are not present, there does not appear to be a "feature[]" of the feature vector [to be] selectively turned on or off." Furthermore, Petitioner does not explain how "features of the feature vector [would be] selectively turned on or off" based on whether "a value" of the data point "is within a predetermined range" because Petitioner relies on Mori's teachings relating to defining the feature vector itself. We, therefore, agree with Patent Owner (Prelim. Resp. 44–50) that Petitioner has not shown that Mori teaches or

suggests a classification method in which one or more features of a feature vector generated from a plurality of static data points are selectively turned on or off based on the value of a static data point.

For these reasons, we find that Petitioner has not shown that Ideses or Mori teaches or suggests claim element 1.e, which recites “generating a feature vector from the plurality of static data points” and “wherein one or more features of the feature vector are selectively turned on or off based . . . on evaluation of whether a value of one of the plurality of static data points is within a predetermined range.” Petitioner’s arguments that claim 1 would have been obvious over Ideses and Mori (Pet. 83–86) do not resolve the deficiency. We also find that Petitioner has not remedied the deficiency in its arguments for independent claims 10, 15, 18, 32, and 46, which have the same recitation. Petitioner also has not remedied the deficiency in its arguments for the dependent claims, each of which depends, directly or indirectly, from independent claim 1, 10, 15, 18, 32, or 46.

Accordingly, we determine that Petitioner has not shown that it is more likely than not that (1) claims 1–59 are unpatentable as obvious over Ideses; (2) claims 1–6, 8–24, 26, 27, 29, 30, 32–38, 40, 41, 43, 44, 46–52, 54, 55, 57, and 58 are unpatentable as obvious over Mori; and (3) claims 1–59 are unpatentable as obvious over the combination of Ideses and Mori.

E. Indefiniteness of Claims 1–59

Petitioner asserts that:

The boundaries of independent claims 1, 10, 15, 18, 32, and 46 are unclear, because the term “the static data points” lacks a reasonably certain antecedent basis. After reciting different types of static data points, the claim language recites “*the* static data points” without providing clarity as to whether “the static data points” refers to the “identified” or “plurality of” static data points.

Pet. 86. Petitioner argues the “specification provides no guidance on how to resolve the ambiguity created in the claim language.” *Id.* at 88.

Patent Owner argues that Petitioner’s argument “fails because the claim language, read in context and in light of the intrinsic record, plainly conveys that ‘the static data points’ analyzed during the classification stage are the ‘plurality of static data points’ extracted from the executable being evaluated during the classification stage.” Prelim. Resp. 52. Patent Owner argues the claim language “alone strongly suggests that ‘the static data points’ refers to ‘the plurality of static data points’ of the classification stage because the two phrases appear in the same clause” and in the context of the claim language “it would be unreasonable to conclude that ‘the static data points’ do *not* refer to the ‘plurality of static data points’ recited only six words earlier in the same clause, and instead, refer to the ‘identified static data points’ last recited three clauses earlier.” *Id.* at 53. Patent Owner also cites to the Specification and the prosecution history in support of its position. *Id.* at 54 (citing Ex. 1001, 1:36–38, 11:60–12:10, 12:43–13:11; Ex. 1002, 79, 121, 132).

A claim is indefinite and therefore unpatentable if it, “read in light of the specification delineating the patent, and the prosecution history, fail[s] to inform, with reasonable certainty, those skilled in the art about the scope of the invention.” *Nautilus, Inc. v. Biosig Instruments, Inc.*, 572 U.S. 898, 901 (2014); *see id.* at 908 (“[D]efiniteness is to be evaluated from the perspective of someone skilled in the relevant art.”); USPTO Memorandum on the Approach to Indefiniteness Under 35 U.S.C. § 112 in AIA Post-Grant Proceedings (Jan. 6, 2021) , available at <https://www.uspto.gov/sites/default/files/documents/IndefinitenessMemo.pdf>. Here, Petitioner does not explain adequately why a person having ordinary skill in the art would have

understood “the static data points” to derive antecedent basis from the different term, “the identified static data points” recited in the “associating” step. Petitioner’s indefiniteness argument does not address persuasively the recitations in claim 1 of “extracting . . . *a plurality of static data points*” and “generating a feature vector from *the plurality of static data points*” (emphasis added). In particular, as Patent Owner points out, the term “the plurality of static data points” is recited six words before “the static data points” in the same “generating” step. Petitioner does not dispute that the first recitation of “the plurality of static data points” derives antecedent basis from “a plurality of static data points” recited in the “extracting” step.

Furthermore, the Specification is consistent with Patent Owner’s argument. *See, e.g.*, Ex. 1001, 1:36–38 (describing that analyzing the “executable file comprises applying a classifier to the plurality of static data points extracted from the executable file”), 12:43–13:11 (describing that “generation of the feature vector may comprise selectively setting features of the learning classifier based on the one or more of the static data points extracted”). Also consistent with Patent Owner’s argument, during prosecution, Applicant pointed to some of the same portions of the Specification relied on here to overcome rejections on the bases of written description and enablement. Ex. 1002, 79, 121, 132.

We also have considered the testimony of Dr. Tsudik (Ex. 1004 ¶¶ 409–412)⁵ that the term “the static data points” renders the challenged claims indefinite. Dr. Tsudik’s testimony is substantively the same as

⁵ The Petition cites to paragraphs 397–408 of Dr. Tsudik’s Declaration (Pet. 86), but his testimony regarding this issue is found in paragraphs 409–412 (Ex. 1004 ¶¶ 409–412). We treat this as a harmless error.

Petitioner's contentions and is not adequate for the same reasons discussed above.

Accordingly, we determine that Petitioner has not shown that it is more likely than not that claims 1–59 are indefinite.⁶

F. Indefiniteness and Lack of Written Description of Claims 8, 14, 17, 29, 43, and 57

Petitioner asserts that:

the string claims [claims 8, 14, 17, 29, 43, and 57] require the “static data points” extracted from the executable file to be character strings, but the independent claims on which they rely require those same data points to have a numeric “value” that can be “within a predetermined range.” A character string does not have a numerical value; thus, the string claims are internally inconsistent and nonsensical, and cannot provide reasonable certainty about the scope of the invention.

Pet. 88–89. Petitioner additionally asserts that “there is no written description in the specification of how character strings can fall within a numeric ‘range,’” and “no disclosure, of non-numeric ranges that can be used to decide whether features in the feature vector are turned on or off.” *Id.* at 94.

Patent Owner argues that Petitioner “makes three incorrect assumptions: (1) that the claims require determining whether a ‘character string’ is ‘within a . . . range,’ (2) that only numeric values can be ‘within a . . . range,’ and (3) that a ‘character string’ cannot be a numeric value.” Prelim. Resp. 56. Patent Owner also argues that claims 8, 14, 17, 29, 43, and 57 have adequate written description even if the claims “require evaluating

⁶ We note that our determination is consistent with the district court’s decision, which found the claims “[n]ot indefinite” due to the term “the static data points.” *See* Ex. 2003, 19.

whether a ‘character string’ falls within a ‘range.’” *Id.* at 58 (citing Ex. 1001, 7:43–52, Table 1.1). Patent Owner also argues that the Specification supports its argument that ranges are not limited to numeric ranges. *Id.* at 59 (Ex. 1001, 7:48–52, 7:54–56).

We agree with Patent Owner that the claims recite evaluating “whether *a value* of one of the plurality of static data points is within a predetermined range” (emphasis added) and the claims do not require “that the ‘value’ is the ‘predefined character string.’” Prelim. Resp. 56. Claim 8, for example, recites “wherein the plurality of static data points *represent* predefined character strings” (emphasis added). Petitioner’s argument that “the ‘static data points’ . . . [are required] to be character strings” and “those same data points [are required] . . . [to] be ‘within a predetermined range’” (Pet. 88–89) omits the recitation of “a value” of one of the plurality of static data points in the independent claims and also omits the recitation that the static data points “represent” predefined character strings in the dependent claims having the string recitations. Petitioner does not explain adequately in light of the correct recitations in claims 8, 14, 17, 29, 43, and 57 why a person having ordinary skill in the art would have lacked reasonable certainty about the scope of the invention.

Petitioner argues that its indefiniteness argument is supported by “distinct categories” disclosed in the Specification. Pet. 89. Patent Owner argues that the Specification describes embodiments in which “a ‘static data point’ *both* represents a predefined character string and has a separate numeric value.” Prelim. Resp. 56 (citing, e.g., Ex. 1001, 7:48–52, 8:39–42); *see also id.* at 57 (arguing that the Specification confirms Patent Owner’s position by “equating the term ‘string’ with a ‘byte sequence’”) (citing Ex. 1001, Table 1.1.). On balance, we find Patent Owner’s argument is

supported by the '869 patent Specification, whereas Petitioner's description of the Specification is too narrow in light of the actual disclosures. The '869 patent Specification describes that its categories "are not limited to" the particular examples described and references a category of "Strings/Byte sequences." Ex. 1001, 6:2–5, Table 1.1. The Specification also describes that "processing and encoding" may vary "depending on the identified categories and/or type" of static data points collected. *Id.* at 8:8–16; *see also id.* at 8:48–50 (describing transforming strings of data "into hex code representation"). We also find persuasive Dr. Malek's testimony that a person having ordinary skill in the art would have known of conversion of character strings into numeric values. Ex. 2001 ¶¶ 100–103.

We find Petitioner's written description argument deficient for many of the same reasons discussed, including that Petitioner's position is premised on the same inaccurate description of the claim language discussed above. *See, e.g.*, Pet. 94. We, instead, are persuaded by Patent Owner (Prelim. Resp. 58–59) that the '869 patent Specification provides written description support for the string limitations. The '869 patent Specification describes a category of "Strings/Byte sequences" (Ex. 1001, 5:2–5, Table 1.1), consistent with Patent Owner's position.

We also have considered the testimony of Dr. Tsudik (Ex. 1004 ¶¶ 413–425) with respect to Petitioner's arguments that claims 8, 14, 17, 29, 43, and 57 are indefinite and lack written description. Dr. Tsudik's testimony is very similar to Petitioner's contentions and is not adequate for the same reasons discussed above.

Accordingly, we determine that Petitioner has not shown that it is more likely than not that claims 8, 14, 17, 29, 43, and 57 are indefinite or lack written description.

IV. CONCLUSION

Petitioner has not demonstrated that it is more likely than not that the challenged claims of the '869 patent are unpatentable.⁷

V. ORDER

In consideration of the foregoing, it is hereby:

ORDERED that the Petition is *denied*, and no trial is instituted.

⁷ Because we deny the Petition on the merits, we do not reach Patent Owner's request that we exercise our discretion to deny the Petition based on the state of the related litigation. *See* Prelim. Resp. 3–21.

PGR2023-00031
Patent 11,409,869 B2

For PETITIONER:

Benjamin Hershkowitz
GIBSON, DUNN & CRUTCHER LLP
bhershkowitz@gibsondunn.com

FOR PATENT OWNER:

Brian Eutermoser
Russell E. Blythe
Mikaela Stone
KING & SPALDING LLP
beutermoser@kslaw.com
rblythe@kslaw.com
mikaela.stone@kslaw.com