

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

SAMSARA INC.,
Petitioner

v.

MOTIVE TECHNOLOGIES, INC.,
Patent Owner

Case IPR2026-00108
U.S. Patent No. 12,062,243

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 12,062,243**

Mail Stop “PATENT BOARD”

Patent Trial and Appeal Board
U.S. Patent & Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	THE '243 PATENT	1
	A. Overview	1
	B. Level of Ordinary Skill in the Art	2
	C. Claim Construction	2
III.	IDENTIFICATION OF THE CHALLENGE (37 C.F.R. § 42.104(b))	3
	A. Prior Art	3
	B. Grounds	4
IV.	OVERVIEW OF THE PRIOR ART	4
	A. Chun	4
	B. Chao	5
	C. Shanmugamani	6
	D. Zhengyang	6
	E. He	7
V.	GROUND 1: CHUN IN VIEW OF CHAO RENDERS OBVIOUS CLAIMS 1, 2, 6-14, 18-20, AND 24-33.	8
	A. Independent Claim 1	9
	1. [1Pre]	9
	2. [1A]	9
	3. [1B] and [1C]	10
	4. [1D]	12
	a. Chun	12
	b. Chao	12
	c. Motivation to Combine (MTC)	13
	5. [1E]	15
	6. [1F]	16
	7. [1G]	17
	a. Chun	17

b.	Chao	18
c.	MTC.....	19
8.	[1H]	21
a.	Chun.....	21
b.	Chao	22
c.	MTC.....	22
B.	Claim 2	23
1.	Chun	23
2.	Chao	24
3.	MTC	24
C.	Claim 6	25
1.	Chun	25
2.	Chao	26
3.	MTC	26
D.	Claims 7 and 8.....	27
1.	Chun	27
2.	Chao	28
3.	MTC	28
E.	Claim 9	29
F.	Claim 10	31
G.	Claim 11	31
H.	Claim 12	32
I.	Claims 13, 14, 18-20, 24, and 25	33
J.	Independent Claim 26	34
K.	Claim 27	35
L.	Claim 28	35
M.	Claim 29	36
N.	Claims 30-33	36

VI.	GROUND 2: CHUN IN VIEW OF CHAO AND FURTHER IN VIEW OF SHANMUGAMANI RENDERS OBVIOUS CLAIMS 3-5, 10, 15-17, AND 21-23.	37
A.	Claim 3	38
1.	Chun-Chao	38
2.	Shanmugamani.....	39
3.	MTC	41
B.	Claim 4	42
1.	Chun-Chao	42
2.	Shanmugamani.....	42
3.	MTC	46
C.	Claim 5	47
1.	Chun-Chao	47
2.	Shanmugamani.....	47
3.	MTC	51
D.	Claims 15-17 and 21-23	51
VII.	GROUND 3: ZHENGYANG IN VIEW OF HE RENDERS OBVIOUS CLAIMS 1-6, 9-18, AND 20-24.	53
A.	Independent Claim 1	55
1.	[1Pre].....	55
2.	[1A]	55
3.	[1B] and [1C]	55
4.	[1D]	57
5.	[1E].....	58
6.	[1F].....	59
a.	Zhengyang	59
b.	He.....	59
c.	MTC.....	60
7.	[1G]	61
a.	Zhengyang	61

b.	He and MTC	62
8.	1[H]	62
B.	Claim 2	63
C.	Claim 3	65
D.	Claim 4	66
E.	Claim 5	68
1.	Zhengyang.....	68
2.	He	68
F.	Claim 6	70
G.	Claim 9	72
1.	Zhengyang.....	72
2.	He	72
3.	MTC	73
H.	Claim 10	73
I.	Claim 11	74
J.	Claim 12	75
K.	Claims 13-18 and 20-24	76
VIII.	GROUND 4: ZHENGYANG IN VIEW OF HE AND FURTHER IN VIEW OF CHAO RENDERS OBVIOUS CLAIMS 7-8, 19, 25-33.	76
A.	Claims 7, 8, 19 and 25	76
1.	Zhengyang-He.....	77
2.	Chao	78
3.	MTC	79
B.	Claims 26-33	80
IX.	NO OBJECTIVE INDICIA OF PATENTABILITY	81
X.	MANDATORY NOTICES	82
A.	Real Party in Interest	82
B.	Related Matters.....	82
A.	Lead and Backup Counsel.....	82

XI.	GROUND FOR STANDING.....	83
XII.	CONCLUSION.....	83

PETITIONER'S EXHIBIT LIST

Exhibit	Description
1001	U.S. Patent No. 12,062,243 to Hassan <i>et al.</i> ("Hassan")
1002	Prosecution History of U.S. Patent No. 12,062,243
1003	Declaration of Dr. Trevor Darrell
1004	<i>Curriculum Vitae</i> of Dr. Trevor Darrell
1005	Chun, Sehyun, et al. "NADS-Net: A nimble architecture for driver and seat belt detection via convolutional neural networks." Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops. 2019.
1006	U.S. Publication No. 2020/0242381 to Chao <i>et al.</i> ("Chao")
1007	Chinese Publication No. CN111860253 to Dongguan Zhengyang Electronic Mechanical Co ltd ("DZE")
1008	He, K., Gkioxari, G., Dollar, P., and Girschick, R. "Mask R-CNN" IEEE International Conference on Computer Vision (2017)
1009	Caruana, Rich. "Multitask Learning." Machine Learning 28 (1997): 41-75.
1010	Shanmugamani, R. Deep Learning for Computer Vision (2018)
1011	P. Smith, M. Shah and N. da Vitoria Lobo, "Determining driver visual attention with one camera," in IEEE Transactions on Intelligent Transportation Systems, vol. 4, no. 4, pp. 205-218, Dec. 2003, doi: 10.1109/TITS.2003.821342.
1012	Baheti, Bhakti & Gajre, Suhas & Talbar, Sanjay. (2018). Detection of Distracted Driver Using Convolutional Neural Network. 1145-11456. 10.1109/CVPRW.2018.00150.
1013	M. S. Majdi, S. Ram, J. T. Gill and J. J. Rodríguez, "Drive-Net: Convolutional Network for Driver Distraction Detection," 2018 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI), Las Vegas, NV, USA, 2018, pp. 1-4, doi: 10.1109/SSIAI.2018.8470309.
1014	U.S. Patent No. 11,176,384 B1 to Yang <i>et al.</i> ("Yang")
1015	U.S. Patent No. 12,347,177 B2 to Li <i>et al.</i> ("Li")

Exhibit	Description
1016	R. Ranjan, V. M. Patel and R. Chellappa, "HyperFace: A Deep Multi-Task Learning Framework for Face Detection, Landmark Localization, Pose Estimation, and Gender Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 41, no. 1, pp. 121-135, 1 Jan. 2019, doi: 10.1109/TPAMI.2017.2781233
1017	Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).
1018	Girshick, R. (2015). Fast R-CNN. In Proceedings of the IEEE international conference on computer vision (pp. 1440-1448).
1019	Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE transactions on pattern analysis and machine intelligence, 39(6), 1137-1149.
1020	Li, M., Chen, S., Chen, X., Zhang, Y., Wang, Y., & Tian, Q. (2019). Actional-structural graph convolutional networks for skeleton-based action recognition. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 3595-3603).
1021	Christoph Maurer, Integration and Deployment of Machine Learning Models (Master's Thesis, Graz University of Technology, Feb. 2021)
1022	Umar Zakir Abdul, Hamid; et al. (2016). "Current Collision Mitigation Technologies for Advanced Driver Assistance Systems-A Survey". PERINTIS eJournal. 6 (2).
1023	Bojarski, M., Yeres, P., Choromanska, A., Choromanski, K., Firner, B., Jackel, L., & Muller, U. (2017). Explaining how a deep neural network trained with end-to-end learning steers a car. arXiv preprint arXiv:1704.07911.
1024	G. Gkioxari, B. Hariharan, R. B. Girshick, and J. Malik. R-cnns for pose estimation and action detection. CoRR, abs/1406.5212, 2014
1025	LeCun, Y., Bengio, Y., & Hinton, G. (2015). "Deep learning." Nature, Vol. 521 (pp. 436-444).
1026	Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

Exhibit	Description
1027	Goodfellow, I., Bengio, Y. & Courville, A. (2016). Deep Learning.
1028	He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving Deeping into Rectifiers: Surpassingb Human-Level Performance on ImageNet Classification.
1029	Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-Excitation Networks. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 7132-7141).
1030	He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition.
1031	Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). ImageNet Classification with Deep Convolutional Neural Networks.
1032	Abouelnaga, Y., Eraqi, H., & Moustafa, M. (2017). Real-time Distracted Driver Posture Classification. In 32 nd Conference on Neural Information Processing Systems (NIPS 2018), Montreal, Canada.
1033	Kose, N., Kopuklu, O., Unnervik, A., & Rigoll, G. (2019). Real-Time Driver State Monitoring Using a CNN Based Spatio-Temporal Approach.
1034	Braunagel, C., Stolzmann, W., Kasneci, E., & Rosensteil, W. (2017). Driver Activity Recognition in the Context of Conditionally Autonomous Driving. In 2015 IEEE 18th International Conference on Intelligent Transportation Systems (pp. 1652-1657).
1035	Engström, J., et al. 2019. A conceptual framework and taxonomy for understanding and categorizing driver inattention.
1036	Choi, I., & Kim, Y. (2015). Head Pose and Gaze Direction Tracking for Detecting a Drowsy Driver. In Applied Mathematics & Information Sciences 9, No. 2L (pp. 505-512).
1037	<i>Motive Techs., Inc. v. Samsara Inc.</i> , Case No. 24-cv-00902-JD (NDCA), Second Amended Complaint
1038	Simonyan, K. & Zisserman A. 2015. Very Deep Convolutional Networks For Large-Scale Image Recognition.
1039	Maurer, C., Integration and Deployment of Machine Learning Models. Graz University of Technology Master's Thesis. (2021).

Exhibit	Description
1040	Boureau et al. 2010. A Theoretical Analysis of Feature Pooling in Visual Recognition, Proceedings of the 27th International Conference on Machine Learning.
1041	Declaration of Dr. Sylvia Ellis-Hall

I. INTRODUCTION

U.S. Patent No. 12,062,243 offers no technical advance. It adopts the same backbone-plus-heads CNN (convolutional neural network) template long used in ADAS (advanced driver assistance system) driver-monitoring: a shared backbone, multiple task heads, supervised loss minimization, and storing a trained model for inference. Ground 1 (Chun-Chao) shows this architecture and storage were conventional. Ground 2 (Chun-Chao-Shanmugamani) shows the claimed detection and pose heads were routine. Ground 3 (Zhengyang-He) confirms modular heads were predictably added to a shared backbone. Ground 4 (Zhengyang-He-Chao) shows no novelty in action-based outputs. The claims redeploy known CNN components warranting institution.

II. THE '243 PATENT

A. Overview

The '243 patent describes a well-known, multitask training pipeline. It frames distracted-driver detection not as a simple classifier, but as a multitask, computer-implemented architecture that trains a backbone CNN on labeled in-cabin driver images and then deploys the trained model in-vehicle for real-time distraction classification. It broadens “distraction” beyond mobile-phone use to include drinking, smoking, talking to passengers, interacting with controls, leaning out of the cabin, and other secondary behaviors, and the claims recite learning those signals jointly. It uses a shared backbone feature extractor coupled to

multiple task-specific prediction heads, including a distraction-classification head, object-detection head, and pose-estimation head. During training, labeled driver images are input into the network, each head computes a loss, and those head losses flow through the backbone to update backbone parameters. After convergence, the backbone and head parameters are stored as the “distraction classification model” and used to classify distraction states in real time on a dash-mounted device. EX1003, ¶¶23-39

B. Level of Ordinary Skill in the Art

A person of ordinary skill in the art (“POSITA”) at the time of the alleged invention would have had a bachelor’s in computer science with a focus on computer/machine vision, and at least two years of experience with each of: (1) machine-learning (including neural-network) methods as applied to machine/computer vision and (2) classical machine/computer vision algorithms (e.g., R-CNN, Mask R-CNN, Fast R-CNN, and SSD). EX1003, ¶86. Additional education could serve as a substitute for experience. *Id.* The Petition is supported by the Declaration of Dr. Trevor Darrell. EX 1003, ¶1-22

C. Claim Construction

Petitioner construed the terms according to their ordinary and customary meanings. *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312-13 (Fed. Cir. 2005).

III. IDENTIFICATION OF THE CHALLENGE (37 C.F.R. § 42.104(b))

A. Prior Art

The Grounds rely on the following prior art, not applied during prosecution.

EX1003, ¶¶75-85:

Reference	Exhibit(s)	Date	Prior Art
Chun	EX1005	Published Feb. 2, 2020	§102(a)(1)
Chao	EX1006	Filed March 26, 2020; Published July 30, 2020	§102(a)(1)-(2)
Zhengyang	EX1007	Filed July 10, 2020; Published October 30, 2020	§102(a)(1)-(2)
He	EX1008	Published June 5, 2018	§102(a)(1)
Shanmugamani	EX1010	Published February 23, 2018	§102(a)(1)

Petitioner has submitted a declaration from Dr. Hall-Ellis—a librarian with over forty years of professional cataloging and bibliographic-verification experience—to confirm the public accessibility and authenticity of Chun, He (Mask R-CNN) and Shanmugamani. *See* EX1041. As explained in her declaration, the Chun paper was published in the Proceedings: 2019 International Conference on Computer Vision and was publicly available no later than February 2, 2020. EX1041, ¶¶53-59. He was published in the IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 42, Issue 2, and was publicly available no later than June 5, 2018, consistent with the date reflected on IEEE Xplore. EX1041, ¶¶46-52. Shanmugamani was issued by Packt in 2018 and was publicly available

no later than February 23, 2018. EX1041, ¶¶67-72. Petitioner relies on Zhengyang, originally published in Chinese. A complete copy of the original Zhengyang document, together with a certified English translation and the translator's certificate of accuracy, is provided in a single combined Exhibit 1007.

B. Grounds

Ground	Basis	Reference(s)	Claims
1	§103	Chun and Chao	1, 2, 6-9, 11-14, 18-20, and 24-33
2	§103	Chun, Chao, and Shanmugamani	3-5, 10, 15-17, and 21-23
3	§103	Zhengyang and He	1-6, 9-18, 20-24
4	§103	Zhengyang, He and Chao	7-8 19, 25-33

IV. OVERVIEW OF THE PRIOR ART

Detecting distracted driving using a multi-task CNN with a backbone and prediction heads for object detection, pose estimation, and distraction classification was well known. EX1003, ¶40.

A. Chun

Chun teaches (National Advance Driver Simulator) NADS-Net, a lightweight CNN for in-vehicle monitoring because “most accidents are caused by distracted driving.” EX1005, 1. Subjects were recorded in-cabin (dual cameras&IR) performing distraction behaviors such as “drinking, using smart phones, yawning,” and the frames were manually annotated (seatbelt masks&MS-COCO keypoints). EX1005, 3-4, FIGs. 1-2. NADS-Net uses a ResNet-50 FPN

backbone with multiple heads (keypoint, PAF, seatbelt). EX1005, 5, FIG. 3.

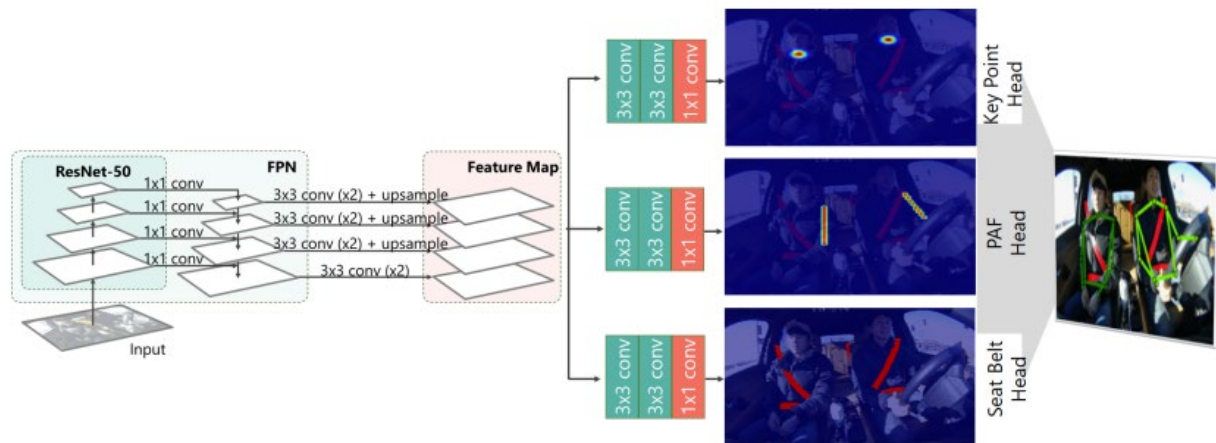


Figure 3: NADS-Net architecture.

EX1005, FIG. 3.

Each head uses “two 3×3 convolutions and a 1×1 convolution to predict a pixel-wise probability.” EX1005, 5. The trained model is then used for IVMS (in vehicle monitoring system), which “play a critical role” in mitigating accidents. EX1005, 1-2. This is standard supervised CNN training (joint loss $\rightarrow$ backbone updates) and storage for inference. EX1003, ¶¶41-48.

B. Chao

Chao teaches monitoring occupants and predicting categorical distraction states (e.g., “Drink,” “Call,” “Text”). EX1006, ¶¶[0098], [0101], [0113]. Features are extracted by a 2D CNN backbone and fed to a spatio-temporal (ST) classifier. Chao teaches supervised training: ST outputs are used as “positive or negative feedback,” and the CNN is “configured to be further trained based on the

feedback.” EX1006, ¶¶[0098], [0041], [0043], [0046]. After training, weights are stored in memory. EX1006, ¶[0054]. This is a conventional backbone+head CNN trained end-to-end and stored for inference. EX1003, ¶¶49-53.

C. Shanmugamani

Shanmugamani teaches how to build deep-learning CV (cross-validation) models for “classification, detection, retrieval, segmentation, generation, captioning, and video classification,” including autonomous driving. EX1010, 1-5. A neural network is “a collection of perceptrons and activation functions,” and CNNs use convolution, pooling, batch-norm, activation, FC (full-connected) layers, and backpropagation to “detect features or patterns in images.” EX1010, 7, 18. For object detection, Shanmugamani describes R-CNN (region-based CNN), Fast/Faster R-CNN, SSD (single-shot multibox detector), YOLO (you only look once), and TensorFlow implementation with pre-trained COCO (common objects in context) models and retraining. EX1010, 109-123, 123-126. Shanmugamani also teaches pose estimation as localization of body joints. EX1010, 110-119; EX1003, ¶¶54-64.

D. Zhengyang

Zhengyang teaches a unified, end-to-end multi-task CNN for driver-attribute recognition. Prior approaches used multiple separate models; Zhengyang instead “adopt[s] a multi-task network model to simultaneously recognize multiple

features.” EX1007, ¶¶[0003], [0005]. Labeled driver images are “input into the multi-task main network to obtain a feature map... [then] input into respective attribute recognition modules.” EX1010, ¶¶[0020]-[0025]. RGB (red-green-blue) frames feed a backbone; Zhengyang’s OSME module creates four feature maps routed to separate heads. EX1010, ¶[0069]. Loss minimization updates parameters until convergence, then weights are stored in a computer-readable medium. EX1010, ¶¶[0022], [0090], [0044]-[0045]; EX1003, ¶¶65-70.

E. He

He teaches a multi-task CNN with a shared ResNet-FPN (feature pyramid network) backbone and multiple heads, extending Faster R-CNN by adding a mask head and a keypoint (pose-estimation) head. EX1008, 7-8. For pose, He states: “Our framework can easily be extended to human pose estimation...[we] adopt Mask R-CNN to predict K masks,” implemented as “eight 3×3 512-d conv layers followed by a deconv layer and $2\times$ bilinear upscaling.” *Id.* The head outputs keypoint heatmaps, trained via cross-entropy, and uses joint backpropagation so losses from the pose head update both the head and the shared backbone end-to-end. EX1003, ¶¶71-74.

V. GROUND 1: CHUN IN VIEW OF CHAO RENDERS OBVIOUS CLAIMS 1, 2, 6-14, 18-20, AND 24-33.

Chun in view of Chao discloses each limitation of the independent claims (1, 13, 20, 26, 30), and the dependent claims fall for the same reasons. EX1003, ¶¶88-89, 97. A POSITA would have combined Chun and Chao because both address distracted-driving detection using multi-task CNNs and teach compatible shared-backbone architectures. EX1003, ¶¶90. Chun provides the backbone and multiple heads, and Chao supplies the express teaching of standard state-classification and loss-minimizing parameter update. EX1003, ¶¶91. Chao explains the ST model operates over “features extracted from images by the 2D CNN,” EX1006, ¶[0101], which are feature maps of the same type produced by Chun’s backbone, so routing Chun’s shared features into Chao’s classification head would require no architectural redesign. Minimizing loss during training was routine and Chao expressly teaches the ST model “provides feedback to further train the 2D CNN,” EX1006, ¶¶[0043], [0098], which a POSITA recognizes as standard backprop-based parameter update. Both references seek improved driver-state accuracy in-vehicle and use the same standardized backbone/head paradigm. EX1003, ¶¶92-95. A POSITA therefore would have reasonably expected success, and the combination yields the claimed subject matter as a predictable application of known CNN practice. EX1003, ¶¶96.

A. Independent Claim 1

1. [1Pre]

Chun discloses the preamble of claim 1. Chun describes training a CNN called NADS Net that “utilizes the feature pyramid network backbone with multiple detection heads to achieve optimal performance for driver/passenger state detection tasks.” EX1005, 1. That training for “2D driver/passenger pose estimation and seat belt detection” is a method, as recited in claim 1. EX1003, ¶98.

2. [1A]

Chun discloses [1A]. Chun describes capturing in-vehicle frames and annotating them so “annotators were instructed to segment all visible seat belts with a binary mask and to mark x and y pixel coordinates of body key points” under the MS COCO protocol. EX1005, 3-4, FIG. 1. These annotations constitute labeled examples identifying seat-belt regions and body key points.

Example annotations are shown below:

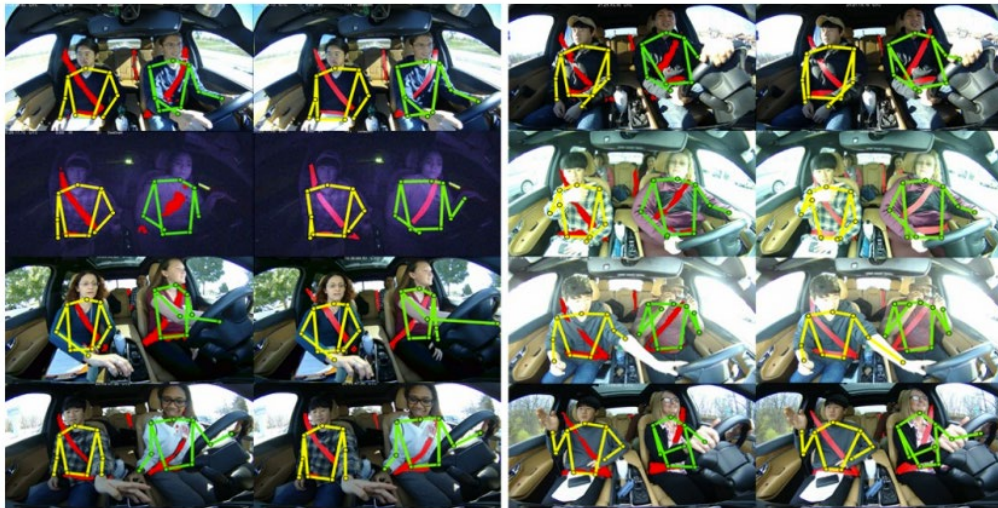


Figure 1: First and third columns in the figure are results produced by NADS-Net and second and fourth columns are the corresponding ground truth annotations.

EX1005, FIG. 1.

Further, the CNN was “pre trained on the MS COCO data set and then transferred to the new driving data set” using Keras with a TensorFlow backend, and trained using labeled key point annotations. EX1005, 5, 7. Chun shows labeled images being input into a neural network for supervised learning, satisfying [1A]. EX1003, ¶¶99-102.

3. [1B] and [1C]

Chun discloses [1B] and [1C]. At the outset, for [1C], a POSITA would understand this limitation is met by a single qualifying head. The phrase “one or more prediction heads” establishes the numerical requirement, and “comprising one or more of” the listed head types indicates any subset satisfies the limitation.

The claim therefore does not require all three head types. A single prediction head of one of the recited types is sufficient. Chun satisfies at least this.

Chun teaches NADS-Net uses a feature-extraction backbone: “a feature pyramid network (FPN) backbone with multiple detection heads to achieve optimal performance for driver/passenger state detection tasks.” EX1005, 1. Figure 3 shows the ResNet-50 FPN backbone generating multi-resolution feature maps that are “further convolved with 1×1 convolutions” to form a shared representation. EX1005, 5-6. These shared backbone features feed multiple downstream prediction heads.

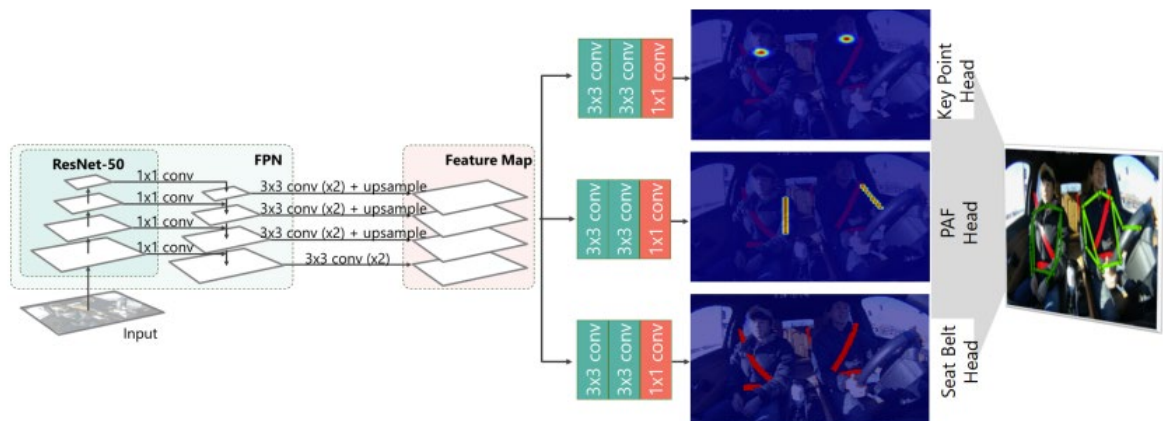


Figure 3: NADS-Net architecture.

EX1005, FIG. 3.

Chun also discloses the prediction heads. NADS-Net “utilizes a FPN backbone with multi-branch detection heads,” including key-point, part-affinity field, and seat-belt segmentation heads. EX1005, 1, FIG. 3. Each head “employs two 3×3 convolutions and a 1×1 convolution” to output pixel-wise probabilities:

joints, joint relationships, or seat-belt mask. EX1005, 5.

Chun discloses a shared backbone and multiple prediction heads, satisfying [1B] and [1C]. EX1003, ¶¶103-112.

4. [1D]

Chun-Chao renders obvious [1D].

a. Chun

Chun states “a vast majority of vehicle accidents are caused by distracted driving behaviors” such as “using smartphones, smoking, drinking, eating, and engaging in conversation,” and trains NADS-Net on those same behaviors (“drinking, using smart phones... yawning... adjusting sun visor”). EX1005, 1,4. A POSITA would view these labeled behaviors as distraction classes. Though Chun does not name a separate distraction head, adding a soft-max classification branch that consumes the same shared feature maps as the pose and seat-belt heads would have been obvious. The new branch would simply consume the shared feature maps already used by Chun’s pose and seat-belt heads and output distraction categories - attentive, phone use, or drinking.

This is also obvious in view of Chao, which expressly discloses a distraction-classification head. EX1003, ¶¶114-119.

b. Chao

Chao describes a CNN extracting spatial features from video frames and a spatio-temporal model classifying driver “states” including “paying attention,

texting, drinking, [and] looking at the radio.” EX1006, ¶¶[0094]-[0098]. A soft-max layer produces class scores (“Action Scores”) representing driver-state categories such as “Drive both hands,” “Drink,” “Call,” and “Text.” EX1006, ¶¶[0101], [0113]. Chao’s architecture, comprising a CNN backbone feeding a classification head, corresponds directly to a distraction-classification prediction head consuming backbone features and outputting categorical labels. EX1003, ¶¶120-125.

c. Motivation to Combine (MTC)

A POSITA would have viewed Chun-Chao as textbook candidates for combination under *KSR*: Chun already provides the exact modular backbone-plus-heads framework into which Chao’s distraction-classification head naturally plugs. *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 417 (2007). Chun proves, in practice, that its FPN backbone can simultaneously feed multiple task-specific heads (pose, seat-belt). Chao teaches a distraction-classification head that consumes the same type of backbone-generated feature maps and outputs categorical distraction state labels. Each reference performs its known function. The combination is simply placing a known distraction-classification head onto a known multi-head backbone that was architected for exactly that type of modular expansion.

The how is obvious. Both references are built on standard multi-task CNN conventions (e.g., shared backbone → specialized heads). Chun already

demonstrates the architectural pattern of branching multiple heads off the same feature maps. Chao's distraction-classification module is implemented in the same way, taking feature maps as inputs and emitting a label. Attaching Chao's classification head to Chun's existing FPN backbone would require no novel architecture, no change to feature extraction, and no redesign of the loss pipeline. A POSITA would connect Chao's head at the same point where Chun attaches its pose/seat-belt heads.

The why is also obvious. Chun expressly identifies distracted-driving behaviors (drinking, phone-use, yawning, leaning on the door) as the problem being addressed. A POSITA would immediately understand that classifying such behaviors into distraction categories is merely the natural next step, i.e., converting the pose/features Chun already extracts into an actual distraction classification. That is exactly what Chao provides. Thus, the motivation is embedded in Chun's own problem statement, not hindsight.

The reasonable expectation of success follows directly. By the priority date, adding a classification head onto a shared FPN backbone was a routine multi-task extension (e.g., Mask-R-CNN and its variants). Chun already proves the backbone can drive multiple heads concurrently. Chao demonstrates a distraction-classification head that uses the same type of backbone features (face/hand/body-related) that Chun already extracts. A POSITA would fully expect that (i) training

the branch, and (ii) receiving categorical distraction labels would work the first time, because both sides of the interface (shared features → classification head) were already conventional and known to interoperate.

There were no technical obstacles. Both use the same conventional CNN primitives. Both operate in the same in-vehicle human-state-inference field. Both aim to solve the same overarching objective, i.e., detect driver distraction. Under *KSR*, that is the exact scenario where the Supreme Court says the combination is obvious: a predictable use of known components, each performing its established function, to yield a straight-line improvement that a POSITA would have been motivated to make and would have reasonably expected to succeed in making.

Chun-Chao renders obvious the distraction-classification prediction head .
EX1003, ¶¶126-136.

5. [1E]

Chun discloses [1E]. Chun teaches an object-detection head for seat-belt identification. Chun explains IVMS detect “correct donning of a seat belt” using vision-based sensing and cites SSD and other detectors. EX1005, 1-3. NADS-Net includes a seat-belt prediction head that “produces a probability density function... indicating the likelihood of a pixel being a seat belt,” which is thresholded to a binary mask. EX1005, 7.

Figure 5 shows an example of this seatbelt detection:



Figure 5: Seat belt detection result (*left*) in comparison with the ground truth (*right*). With human inspection, the prediction result is of good quality as it correctly marks the seat belt area. However, IOU for this particular example was 46%, justifying the need for a better evaluation metric.

EX1005, FIG. 5.

A POSITA would recognize the seat-belt head performs object detection: it identifies the “seat belt” class and its location via a dense probability map, functionally analogous to bounding-box detection in Faster R-CNN or SSD cited by Chun. EX1003, ¶¶137-141.

Chun discloses an object-detection prediction head satisfying [1E].

6. [1F]

Chun discloses [1F]. Chun teaches pose estimation heads within NADS Net. Chun states the key point and part affinity field heads together perform “2D driver/passenger pose estimation.” EX1005, 1. Chun collected in vehicle images of drivers performing behaviors such as “drinking, using smartphones, yawning, sneezing, leaning on the door, adjusting the sun visor” to supplement MS COCO. EX1005, 4. The key point head predicts pixel probabilities for each joint and the

PAF head encodes pairwise joint relationships and the “outputs from the first two heads are used to parse the key point instances into human skeletons.” EX1005, 5.

Chun discloses pose estimation prediction heads of [1F]. EX1003, ¶¶142-145.

7. [1G]

For avoidance of doubt, Petitioner does not contend [1G] requires a “joint loss”; the plain language is satisfied so long as a loss “based on outputs of the one or more prediction heads” is minimized in a way that updates the backbone parameters, including where that loss is derived from a single prediction head. Nonetheless, regardless of interpretation, Chun-Chao renders obvious [1G]. EX1003, ¶¶146-147.

a. Chun

Although Chun does not expressly state “loss minimization,” its TensorFlow multi-head CNN, supervised training, and accuracy/transfer-learning experiments indicate standard loss minimization and backpropagation. EX1003, ¶¶148-149. Chun trains NADS-Net on labeled driver images using key-point, PAF, and seat-belt heads sharing an FPN backbone, reports “training and validation” across datasets, and compares “pretrained” versus “transferred” models, EX1005, 5-7, terminology a POSITA would understand as iterative joint-loss minimization updating shared backbone parameters. EX1003, ¶149. Each head outputs a

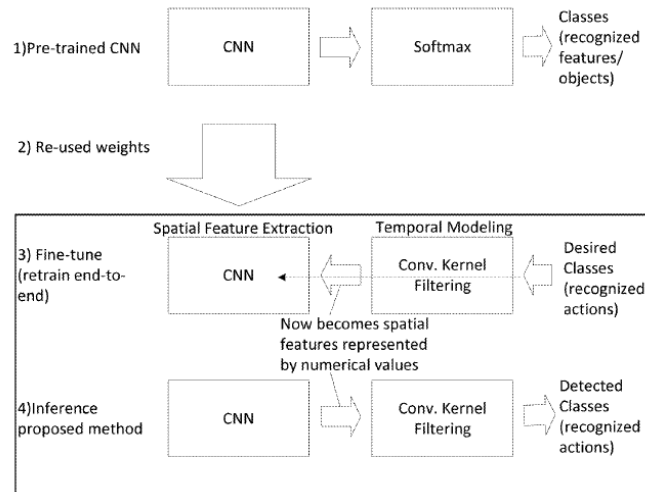
probability map that is compared to ground truth, and the combined task losses are minimized jointly; the conventional multi-task approach used in OpenPose and Mask R-CNN, which Chun cites.

A POSITA would understand Chun to be minimizing a joint loss across the prediction-head outputs to update the shared backbone, and would look to Chao for the implementation details of that joint-loss training. EX1003, ¶¶150-154.

b. Chao

Chao expressly teaches end-to-end supervised training. Chao states the system uses “objective functions (i.e., loss functions)” and “any kind of training principle, such as backpropagation.” EX1006, ¶¶[0041], [0046]. Chao explains that prediction outputs are used “to further train the CNN.” EX1006, ¶[0098]. Figure 8A confirms that the “pre-trained CNN” is then “fine-tune[d] (retrain end-to-end),” meaning the backbone parameters are updated based on final output loss.

FIG 8A



EX1006, FIG. 8A.

Chao teaches backbone parameters are updated based on final classification outputs. Chao states the “2D CNN 802 may change the output ... to represent a plurality of values for the new desired ‘action classes.’” EX1006, ¶[0099]. Chao explains “feedback from the results produced by [the] ST system” is used “to train the CNN” with “backpropagation.” EX1006, ¶¶[0046], [0098]. A POSITA would view this as standard supervised learning in which prediction-head losses backpropagate to update the backbone. EX1003, ¶¶155-162.

c. MTC

A POSITA would recognize that applying Chao’s loss-minimization to Chun’s multi-head CNN is simply standard practice. The norm was to jointly optimize backbone and heads by minimizing task-output loss, and Chao says so

explicitly. Chun suggests this conventional multitask training, even if it omits the loss formula. A POSITA would therefore fill that gap with Chao's explicit teaching because it is the ordinary way to improve CNN accuracy in this field.

A POSITA reading Chun-Chao together would immediately see how to combine them and why it improves accuracy. Chun already uses a shared backbone with multiple heads for driver monitoring and trains them end-to-end in TensorFlow. Chao supplies the missing explicit piece: a loss-minimization routine in which the distraction-classification loss is backpropagated to update those shared backbone weights.

The how is the same standard workflow used before the '243 patent: define a classification loss for the distraction head and let the framework backpropagate it into the shared FPN weights. No architectural change is needed.

The why is straightforward. Chun seeks optimal driver-state accuracy, and loss minimization is the known way to improve it. Chao tells exactly how to implement that loss. A POSITA would apply that routine to Chun because it is the standard and obvious method to optimize weights for distraction recognition and Chun is already structured to benefit

A POSITA would also have held a strong expectation of success. At the time, loss minimization with gradient based weight updates was the default paradigm in deep learning, and TensorFlow already propagated gradients from the

head back into the backbone automatically. No re-architecting or new training logic is required, because the approach is simply the ordinary application of an already established technique to the same architecture and the same task. A POSITA would reasonably expect that adding Chao's explicit loss to Chun's backbone and heads would predictably improve distraction-classification accuracy in the same driver monitoring context.

[1G] represents nothing more than a predictable substitution. EX1003, ¶¶163-171.

8. [1H]

Chun-Chao renders obvious [1H].

a. Chun

Chun teaches a trained, deployable CNN (“more nimble and thus more suitable for in-vehicle monitoring” and “non-invasive, non-obstructive means to monitor and detect in-cabin activities”) which a POSITA would understand requires storing the trained network weights (backbone + heads) for runtime inference. Chun's discussion of comparing “trained only on MS COCO,” “trained only on the new driving dataset,” and “pretrained then transferred,” EX1005, 7, strongly suggests checkpointing finalized models for evaluation and deployment. EX1003, ¶¶173-175. Where Chun suggests it, Chao is explicit.

b. Chao

Chao states “a memory of the one or more memories 104 may store ... a trained system, such as a neural network, or a deep neural network.” EX1006, ¶[0054]. Chao’s disclosed end-to-end training loop, EX1006, ¶¶[0041], [0046], [0098], confirms that after gradient-based updating, the trained network is saved in memory for inference. EX1003, ¶176

c. MTC

A POSITA reading Chun-Chao together would understand how and why to store the trained distraction model. Chun already trains multiple heads and a shared backbone until convergence for in vehicle inference. Chao provides the explicit teaching that once training is complete, learned backbone and head parameters are saved to non-volatile memory for later loading. After convergence, the framework storage primitive is invoked to serialize those parameters to a file or checkpoint, which TensorFlow and PyTorch already provide. Chun must store trained parameters to run in a vehicle, so adding Chao’s storage step is the ordinary final deployment step.

A POSITA would reasonably expect storing the fully trained model to work on first application because serialization and re-loading were routine. Saving optimized backbone weights and prediction-head weights as a single artifact was the standard way to deploy a trained CNN. Nothing in Chun’s architecture or data

flow prevents storage, and nothing in Chao is incompatible with Chun. Both references assume the same CNN paradigm, the same backbone plus heads, and the same field of use. The combination simply applies the known deployment step to the known multi-head model, yielding the predictable result: the same model Chun already trains is now persisted and ready for inference in the vehicle runtime environment. Under *KSR*, that is obvious. It is the ordinary use of a known technique model storage to improve a closely related system that is already designed for deployment. *KSR*, 550 U.S. at 417; EX1003, ¶¶177-183.

B. Claim 2

Chun-Chao renders claim 2 obvious. EX1003, ¶184.

1. Chun

Chun expressly discloses prediction heads made of multiple convolutional layers coupled to a backbone (two 3×3 convolutions and a 1×1 convolution) that output pixel-wise probability maps for key-point, part-affinity, and seat-belt heads. EX1005, 5, FIG. 3; EX1003, ¶¶185-186. Claim 1 shows Chun-Chao renders obvious the distraction-classification head. EX1003, ¶186. Although Chun does not list pooling or fully connected layers, a POSITA would understand convolution plus pooling plus fully connected layers was the standard classifier-head pattern. Adding that branch for distraction classification to Chun's FPN features would have been obvious, as taught in Chao. EX1003, ¶187.

2. Chao

Chao's ST head receives features "extracted from the 2D CNN model" and applies convolutional layers for feature refinement. EX1006, ¶¶[0099], [0106]; EX1003, ¶¶188-190. Chao discloses dimensionality reduction before the fully connected layer ($N_k < N_s$), which a POSITA would recognize as the pooling step. EX1006, ¶[0109]. Chao then applies a fully connected layer that outputs "action scores" identifying distraction states. EX1006, ¶[0107]; EX1003, ¶¶191-192.

3. MTC

A POSITA would understand Chun already supports adding new task heads, and Chao supplies the standard conv then pool then FC pattern for classification heads. The classifier would attach to Chun's shared FPN features so convolution refines features, pooling aggregates them, and the FC layer outputs distraction class scores. No redesign is required. Both references target driver state recognition and use the same modular backbone-plus-heads paradigm. Conv then pool then FC was already the conventional classification head pattern, and adding it to Chun would yield categorical distraction labels using the same backbone features.

A POSITA would have had a strong reasonable expectation of success. Chun's backbone encodes the exact cues Chao uses: face geometry, hand motion, body pose. Attaching a conv-pool-FC branch to those features was a routine extension that would predictably yield accurate categorical distraction predictions.

This is the familiar application of known components in the familiar way to yield a familiar result, exactly the type of combination *KSR* identifies as obvious. *KSR*, 550 U.S. at 417, 421; EX1003, ¶¶193-197.

C. Claim 6

Petitioner is not construing Claim 6 to require that the distraction-classification model must include the “distraction classification prediction head” or that the distraction-classification prediction head produces the distraction-classification tag. Claim 6 only recites that the model receive image data and output a distraction-classification tag, and that is satisfied whether the model uses a dedicated classification head, a shared head, or a single prediction head that performs the classification function. Regardless of the interpretation, Chun-Chao renders obvious claim 6. EX1003, ¶¶198-199

1. Chun

Chun teaches inputting in-vehicle driver images into a multi-task CNN and producing outputs such as pose estimation and seat-belt detection. EX1005, 1, 4-5. Chun shows training on annotated distraction behaviors such as drinking and phone use, and a POSITA would understand the same trained network receives real-time driver images and outputs predictions during inference. EX1005, 4-5.

Adding Chao's classification head to Chun would therefore yield the recited step of inputting image data into a distraction-classification model and receiving a distraction-classification tag. EX1003, ¶¶200-201.

2. Chao

Chao expressly discloses inputting driver images into a CNN based driver monitoring system and receiving a categorical output identifying driver distraction state. EX1006, ¶¶0129, 0136. Chao's architecture uses a shared 2D CNN backbone and a SoftMax classification layer that emits "action scores." EX1006, ¶0107. Chao identifies concrete distraction classes such as "Drink," "Call," "Text," "Makeup," and binary tags such as "FULL_ATTENTION" versus "DRIVER_TEXTING." EX1006, ¶[0113], FIG. 6. A POSITA would understand Chao discloses receiving a discrete distraction-classification tag based on image data. EX1003, ¶¶202-206.

3. MTC

A POSITA would have added Chao's classifier branch to Chun's existing multi-head CNN. Chun already routes driver-image frames through a shared FPN backbone into multiple heads for pose and seat-belt detection, and Chao provides a standard SoftMax classifier head that consumes the same type of backbone features and outputs a categorical distraction label. Attaching Chao's head to Chun's shared feature maps would yield the claimed distraction-classification tag. That is the

same conv-backbone-plus-classification-head pattern that was industry standard before the '243 patent.

A POSITA would also know why. Chun seeks improved distracted driving detection, and Chao supplies the class labels that make the detection actionable. Combining them yields the expected real time interpretable output without altering the backbone or training pipeline. Success would have been expected because both references already run real time inference using the same CNN conventions. Attaching a standard SoftMax branch to existing FPN features was a routine multitask extension requiring no new architecture. Under *KSR* this is the predictable combination of familiar s yielding a predictable result. EX1003, ¶¶207-218.

D. Claims 7 and 8

Chun-Chao renders obvious claims 7 and 8. EX1003, ¶¶219-220, 232-233

1. Chun

A POSITA would understand Chun-Chao discloses generating a distraction-classification tag identifying states such as texting or calling. EX1003, ¶¶221, 234. It would have been obvious to then perform a downstream action based on that tag. Chun states IVMS “play a critical role in preventing and mitigating traffic accidents by alerting the distracted driver and adaptively adjusting the safety mechanisms.” EX1005, 1; EX1003 ¶¶222, 235. A POSITA would view Chun’s

CNN outputs as the perception signals that drive those alerts and adjustments, as taught in Chao.

2. Chao

Chao makes the actuation explicit: after determining the driver's state, the "spatio-temporal analysis modeling scheme" "trigger[s] vehicular actions based on the determined state." EX1006, ¶¶[0022], [0129]; EX1003 ¶¶223, 236. Those actions include "modifying ... acceleration, braking, [or] steering" or "communicating ... notifications via the UIs." EX1006, ¶[0095]. Notifications may be "visual," "sound," or "vibration." EX1006, ¶[0168]; EX1003, ¶¶224, 237. A POSITA would equate "visual" with updating a display and "sound" with an audible alarm; and would view forwarding notifications to other processors, EX1006, ¶¶[0117], [0032], as transmitting the prediction to a remote controller/server. EX1003, ¶¶224, 238-240.

3. MTC

A POSITA would have combined Chun-Chao by routing the distraction-classification tag output from Chao's classifier head (integrated into Chun's multi-head CNN) to the control module disclosed in Chao. EX1006, ¶¶[0095]-[0168]; EX1003, ¶¶225, 241. At runtime, Chun's backbone receives driver images, extracts shared features, and those features are fed to Chao's classifier, which outputs a categorical distraction tag. EX1003, ¶¶225, 241. That tag then conditions the driver

alert or control adjustment using routine IVMS message-passing between the CNN and the controller. EX1003, ¶¶225, 241-242.

Chun already extracts features for distracted behaviors such as “drinking, using smart phones, exercising, yawning, sneezing, leaning on the door, putting hands out the window, [and] adjusting sun visor.” EX1005, 4; EX1003, ¶¶226, 242. Chao identifies nearly the same classes (e.g., “Drink,” “Call,” “Look right,” “Read,” “Text”). EX1006, ¶[0113]; EX1003 ¶¶226, 242. A POSITA would view adding Chao’s ST classification branch to Chun’s backbone as a direct modular extension because both use shared backbone features feeding prediction heads. Chun states IVMS “alert the distracted driver and adaptively adjust the safety mechanisms.” EX1005, 1. Chao discloses triggering vehicular actions based on state. EX1006, ¶[0065]. Combining them yields a predictable result, actionable distraction tags, with expected success. EX1003, ¶¶227-231, 243-245.

E. Claim 9

Chun-Chao renders obvious claim 9. EX1003, ¶246

Chun already provides the shared-backbone architecture generating feature maps for multiple heads. EX1003, ¶¶247-250.

Chao: The claim simply recites that the distraction-classification head outputs a label (undistracted / mobile-usage / other distracted), as Chao already discloses. Chao teaches the ST output is “a vector of NC values” normalized by

SoftMax to form a “probability distribution vector,” and that “each of the classes of the vector” is output as the driver-state label (“Action Score”). EX1006, ¶[0101]; EX1003 ¶¶251-252. Table 1 maps those class labels into categories a POSITA would recognize as undistracted (e.g., “Drive both hands”), mobile usage (e.g., “Call,” “Text”), or other distracted (e.g., “Adjust radio,” “Makeup,” “Reach side”). EX1006, ¶[0113]; EX1003 ¶¶253-255.

TABLE 1

Exemplary list of classes (i.e. states) and corresponding reaction time values													
	Index												
	1	2	3	4	5	6	7	8	9	10	11	12	13
Class	Drive both hands	Switch gear	Talk to passenger	Adjust Radio	Drink	Call left hand	Call right hand	Hair makeup	Reach side	Reach behind	Text left hand	Text right hand	Hands free
Reaction time	2.0 sec	2.1 sec	2.4 sec	3 sec	3 sec	3.2 sec	3.2 sec	3.5 sec	4 sec	4 sec	4.2 sec	4.2 sec	5 sec

EX1006, Table 1.

Chao supplies the explicit SoftMax classification head that consumes those maps and outputs categorical labels. EX1003, ¶256.

MTC: For the reasons given for claims 6-7, a POSITA would have found it obvious to integrate Chao’s classifier branch into Chun’s backbone to output the distraction-state labels recited in claim 9. EX1003 ¶¶256-261.

F. Claim 10

Chun discloses claim 10. EX1003, ¶262. NADS-Net uses a shared FPN backbone with multiple heads, including a seat-belt head that outputs a pixel-probability map thresholded to a binary mask. EX1005, 1-2,7; EX1003, ¶¶263-264. Chun states this provides “more granularity in detection results” to locate the belt. EX1005, 2; EX1003, ¶264. Chun teaches all heads use the same backbone feature vector. EX1005, 2, 5; EX1003 ¶264. A POSITA would understand the seat-belt head receives that backbone vector as its input and performs object-location detection.

Chun discloses the claimed object-detection prediction head. EX1003, ¶264.

G. Claim 11

Chun discloses claim 11. EX1003, ¶265. Chun’s FPN backbone feeds prediction heads for 2D pose estimation, including key-point and PAF heads, trained on distracted-driving gestures such as drinking, phone use, yawning, leaning, and adjusting the sun visor. EX1005, 1, 4; EX1003, ¶266. Chun states the PAF head outputs vector fields capturing pairwise joint relationships. EX1005, 2, 5; EX1003 ¶267. A POSITA would understand the PAF head receives the same backbone feature vector and uses that vector to infer joint orientation. EX1003, ¶267.

Chun discloses a pose-estimation head that detects orientation of body parts from an input vector. EX1003, ¶268.

H. Claim 12

Chun-Chao renders this claim obvious. EX1003, ¶269

Chun presents NADS-Net as an IVMS-deployment model “more nimble and thus more suitable for in-vehicle monitoring” and capable of “non-invasive, non-obstructive” cabin monitoring. EX1005, 1; EX1003, ¶270. Chun’s deployment teaching implies storing trained weights for inference in-vehicle, and shows cabin-mounted cameras, including one “mounted below the rear view mirror” with dashboard IR illumination. EX1005, 4; EX1003 ¶271. Chun suggests deployed storage proximate to dash-region sensing hardware, but may be alleged to be silent on storing in the dash-mounted camera itself, which is explicit in Chao. EX1003 ¶272.

Chao makes that explicit: memory 104 “may store ... a trained system, such as a neural network,” and the camera acquisition device may be “on the dash.” EX1006, ¶¶[0054], [0086], FIG. 5A; EX1003, ¶¶273-275. Chao teaches co-locating camera, processor, and memory in one housing. EX1006, ¶[0258]; EX1003 ¶276.

MTC: A POSITA would have viewed storing the trained model in the dash mounted camera as a routine deployment choice suggested by Chao and consistent

with Chun's deployment context. EX1003 ¶¶277-278. A POSITA would load the trained NADS Net weights into the local memory of the dash mounted camera hardware that Chao discloses. Chun-Chao use CNNs for in vehicle driver monitoring with interior cameras, and storing trained weights in the same housing as the capture device was already routine. Applying Chao's deployment architecture to Chun's trained model is a straightforward engineering step. Chun's goal is real-time distracted driving detection, and Chao discloses storing the trained network in the dash-mounted camera for in vehicle inference. A POSITA would expect success because both references use the same backbone-plus-head CNN paradigm, both already perform real-time inference, and storage of trained weights is task agnostic. EX1003, ¶¶279-281.

I. Claims 13, 14, 18-20, 24, and 25

Claims 13, 14, 18-20, 24, and 25 add no new features. They restate the limitations of claims 1, 2, 6-9, 11, and 12 as computer-readable medium (CRM) or device. EX1003, ¶282. Claim 13 mirrors claim 1 as a non-transitory medium; claims 14, 18, and 19 mirror claims 2, 6, and 7; claim 20 mirrors claim 1 as a processor-and-memory device; and claims 24 and 25 mirror the same refinements of claims 6, 7, 18, and 19. EX1003, ¶¶283-284. A POSITA would understand that the same Chun-Chao method is stored as program instructions or executed on in-vehicle processors. Chao expressly discloses memory storing a trained neural

network and processors running it. EX1006, ¶¶[0092]-[0093]; EX1003, ¶286.

Converting an obvious method into standard software or device form would have been routine. For the same reasons as claims 1, 2, and 6-12, these claims are likewise obvious. EX1003 ¶¶285-288.

J. Independent Claim 26

Claim 26 recites the same subject matter as claim 1, with the same additional features already addressed for claims 6, 7, 8, and 12. EX1003, ¶289. Each component limitation has already been shown obvious in Chun-Chao. EX1003, ¶289. Claim 26[A] is the same dash-mounted storage limitation as claim 12: Chun suggests in-vehicle model deployment and Chao expressly discloses storing the trained network in the dash-mounted camera. EX1003, ¶290. Claim 26[B] is the same backbone-plus-heads architecture and supervised training as claim 1, already taught by Chun-Chao. EX1003, ¶291. Claim 26[C] recites capturing an interior vehicle image using a dash-mounted sensor, which both Chun-Chao teach. EX1003 ¶¶292-294. Claim 26[D] is the same image input and distraction tag output as claim 6, and claim 26[E] is the same downstream action as claims 7 and 8. EX1003 ¶¶295-296. For the same reasons as claims 1, 6-8, and 12, claim 26 is obvious in view of Chun-Chao. EX1003, ¶289.

K. Claim 27

Claim 27 recites substantially the same subject matter as claim 9. Chun-Chao discloses classifying the driver image into categorical labels corresponding to undistracted, mobile usage, or other distracted states using an input vector generated based on images of drivers. EX1003, ¶297.

L. Claim 28

Chun discloses claim 28. Chun's seat belt detection head produces a probability density over the image domain "indicating the likelihood of a pixel being a seat belt," which is then thresholded to produce a binary segmentation mask that identifies the seat belt location relative to the driver. EX1005, 3, 7, FIG. 5; EX1003, ¶298. An example of this seatbelt detection is in Chun Figure 5:



Figure 5: Seat belt detection result (*left*) in comparison with the ground truth (*right*). With human inspection, the prediction result is of good quality as it correctly marks the seat belt area. However, IOU for this particular example was 46%, justifying the need for a better evaluation metric.

EX1005, FIG. 5.

This seat belt detection prediction head provides the exact location of a seatbelt in an image of a driver/passenger using the pixels of the image. A

POSITA would understand this as detecting the location of an object within the image. EX1003, ¶299.

M. Claim 29

Chun describes a feature pyramid backbone with prediction heads for “2D driver / passenger pose estimation,” including key point and PAF heads trained on images of drivers performing gestures such as “drinking, using smart phones, yawning, sneezing, leaning on the door, putting hands out the window, searching floor and the center console, adjusting sun visor, and etc.” EX1005, 1, 4; EX1003, ¶300.

Chun explains that the PAF head produces vector fields that “encode pairwise relationships between body joints” indicating how keypoints are connected. EX1005, 2, 5; EX1003 ¶301. A POSITA would understand that the PAF head detects body part orientation depicted in the image. EX1003 ¶301.

Chun discloses a pose estimation prediction head, detecting orientation of one or more body parts in image data. EX1003, ¶302.

N. Claims 30-33

Claims 30-33 restate the same features of claims 26-29 in device form rather than method. EX1003, ¶303. Claim 30 mirrors claim 26, but recites a dash-mounted camera device with processor and memory, and claims 31-33 mirror claims 27-29 (label classification, object-location detection, and pose-orientation

detection). EX1003, ¶304. A POSITA would understand that the same distraction-classification functionality taught by Chun-Chao is executed by processors in an in-vehicle device. EX1003, ¶305. Chao expressly discloses such modules, including in-vehicle cameras, memory storing a trained neural network, and processors. EX1005, ¶¶[0092]-[0093]; EX1003, ¶306. Recasting the same obvious operations into “device” form adds no inventive distinction. For the same reasons as claims 26-29, claims 30-33 are obvious in view of Chun-Chao. EX1003, ¶307.

VI. GROUND 2: CHUN IN VIEW OF CHAO AND FURTHER IN VIEW OF SHANMUGAMANI RENDERS OBVIOUS CLAIMS 3-5, 10, 15-17, AND 21-23.

Chun in view of Chao and Shanmugamani discloses each added feature of dependent claims 3-5, 10, 15-17, and 21-23. These claims recite routine CNN blocks already used in Ground 1 (convolution stacks, pooling, fully connected layers, bounding box regression, class heads, and pose heads), which were standard TensorFlow/Keras components for object detection and image classification. EX1003 ¶¶308-309.

A POSITA would have combined Chun, Chao, and Shanmugamani because Chun expressly states NADS-Net was implemented in TensorFlow, and Shanmugamani teaches the same off-the-shelf TensorFlow layer patterns for classification, detection, and pose estimation used by Chun and Chao. EX1003, ¶¶310-317. This is not inventive: Shanmugamani provides step-by-step

TensorFlow CNN implementations for autonomous-camera workloads and confirms that the dependent-claim details reflect conventional blocks. EX1003, ¶¶318-319.

A POSITA would have had a reasonable expectation of success, and dependent claims 3-5, 15-17, and 21-23 are obvious for the same reasons as Ground 1. EX1003, ¶¶320-323.

A. Claim 3

Chun-Chao-Shanmugamani renders obvious claim 3 that depends from claim 1. EX1003, ¶324. Chun-Chao renders claim 1 obvious. EX1003, ¶325. Claim 3 simply recites that the object detection head includes a bounding-box regression network to output box coordinates and a class-prediction network to output an object label. EX1003, ¶325.

1. Chun-Chao

Chun already teaches object localization. Chun states the seatbelt head provides “proper/improper use cases judged by the relative position of seat belt to the detected body position.” EX1005, 3. This is object localization of the same type used in bounding box detection as in Faster R-CNN and SSD which Chun cites. EX1003 ¶¶326-327.

Chao likewise teaches determining an object footprint and classifying it. EX1006, ¶¶[0046], [0075], [0083].

A POSITA would therefore understand Chun-Chao's seatbelt head as implemented through bounding box regression followed by class prediction, and Shanmugamani provides explicit disclosure of the dual head object detection structure recited in claim 3. EX1003, ¶¶328-329.

2. Shanmugamani

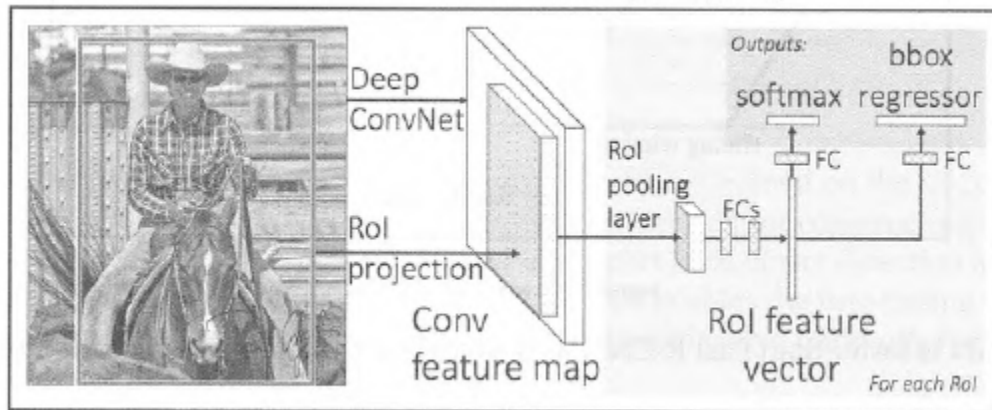
Shanmugamani explains CNNs are “a collection of perceptrons and activation functions” and “CNNs have weights, biases, and outputs through nonlinear activation” trained to “detect features or patterns in images.” EX1010, 7, 18; EX1003, ¶330

For object detection, Shanmugamani states “object detection is the act of finding the location of an object in an image” by finding objects “along with the labels.” EX1010, 109-110. Shanmugamani describes localization as a regression problem where “the bounding box is four numbers” and that there is also a “need to predict the label which is a classification problem.” EX1010, 117.

Shanmugamani confirms “the regression encoder and classification encoder function independently” and a bounding box may be parameterized by “center of the coordinates with the height and width of the bounding box.” EX1010, 119; EX1003, ¶331.

Shanmugamani then provides concrete TensorFlow examples of this exact dual head structure, including R CNN, Fast R CNN, Faster R CNN, SSD, and

YOLO which all output bounding box coordinates and class predictions. EX1010, 120-123. For example, in R CNN, a “linear regression model to predict a closer bounding box is trained with object region proposals” and “the last layer of the CNN is trained with an SVM identifying the object with a no object class.” EX1010, 120. Fast R CNN uses ROI pooling with “one hot encoding with softmax” and a bounding box regressor. EX1010, 121. SSD “simultaneously predicts the object and finds the bounding box.” EX1010, 122-123; EX1003 ¶¶332-336.



EX1010, 121.

These teachings show a bounding box regression network plus a classification network is the standard implementation of object detection in contemporary CNNs. EX1003, ¶¶337-338. A POSITA therefore would understand that claim 3 recites the well-known dual head pattern taught directly in Shanmugamani, including for automotive computer vision. EX1003, ¶339.

3. MTC

A POSITA would have been motivated to combine Chun-Chao-Shanmugamani. EX1003, ¶340. Chun already implements NADS-Net in TensorFlow, uses COCO, and detects seat belts, but does not spell out the standard TensorFlow layer recipe for bounding-box, regression-plus-class prediction. A POSITA using Chun's TensorFlow platform would naturally consult a contemporaneous TensorFlow reference, and Shanmugamani provides exactly that: explicit TensorFlow object-detection implementations using COCO, with bounding-box regressors and class-prediction heads trained end-to-end. EX1003, ¶340.

Chun-Chao already identify seat-belt location and classify object state relative to the driver. Implementing that in the standard TensorFlow form taught by Shanmugamani would require no redesign; it is simply the routine COCO-style bounding-box plus class-prediction pattern. EX1003, ¶340.

And Shanmugamani expressly states the reader “will learn how to train deep learning models for computer vision applications and deploy them on multiple platforms,” including autonomous-vehicle detection. *Id.*, 6. Because Chun already uses TensorFlow, a POSITA would expect straightforward success inserting Shanmugamani's standard TensorFlow object-detection blocks into Chun's TensorFlow seat-belt head. EX1003, ¶¶341-342.

Chun-Chao-Shanmugamani renders obvious claim 3. EX1003, ¶343.

B. Claim 4

Chun-Chao-Shanmugamani renders obvious claim 4. EX1003, ¶344. Chun-Chao-Shanmugamani renders obvious claim 3. EX1003, ¶345. Claim 4 only adds that the bounding box regression network and object class prediction network include hidden layers with convolution, batch normalization, and activation, which are conventional CNN s expressly taught in Shanmugamani.

1. Chun-Chao

Chun discloses multiple convolutional layers in its seat belt head (“two 3×3 convolutions and a 1×1 convolution”). EX1005, 5. A POSITA would understand these as hidden layers. Chao likewise teaches CNNs with input, output, and “one or more hidden layers.” EX1006, ¶¶[0075], [0098]. While Chun-Chao do not enumerate the standard internal sub layers within each hidden layer, Shanmugamani provides those conventional implementation details. EX1003, ¶¶346-347.

2. Shanmugamani

Shanmugamani confirms that bounding box regression and object class prediction networks are conventionally implemented using convolutional, activation, and batch normalization layers. Shanmugamani teaches a neural network “is a collection of perceptrons and activation functions,” where “inputs of x are passed through a hidden layer of perceptrons and summed to the output,” and

“every layer can have one of the activation functions.” EX1010, 11. Activation functions “make neural nets nonlinear” and adjust gradients during training, including sigmoid, tanh, and ReLU. EX1010, 7-9. Shanmugamani teaches batch normalization “increases the stability and performance of neural network training” because it “reduces overfitting and makes the network train faster” by “normalizing the output from a layer with zero mean and a standard deviation of 1.” EX1010, 13.

Shanmugamani then walks through “retrain[ing] a model for our custom dataset,” including dataset preparation, algorithm selection, and fine tuning. EX1010, 126. As an explicit example, Shanmugamani presents a YOLO based pedestrian detection model for self driving applications that includes convolutional layers, activation (tf.nn.leaky_relu), and regularization layers. EX1010, 128-130.

```
def pooling_layer(input_layer, pool_size=[2, 2], strides=2,
padding='valid'):
    layer = tf.layers.max_pooling2d(
        inputs=input_layer,
        pool_size=pool_size,
        strides=strides,
        padding=padding
    )
    add_variable_summary(layer, 'pooling')
    return layer
```

```
def convolution_layer(input_layer, filters, kernel_size=[3, 3],
padding='valid',
                        activation=tf.nn.leaky_relu):
    layer = tf.layers.conv2d(
        inputs=input_layer,
        filters=filters,
        kernel_size=kernel_size,
        activation=activation,
        padding=padding,
        weights_initializer=tf.truncated_normal_initializer(0.0, 0.01),
        weights_regularizer=tf.l2_regularizer(0.0005)
    )
    add_variable_summary(layer, 'convolution')
    return layer

def dense_layer(input_layer, units, activation=tf.nn.leaky_relu):
    layer = tf.layers.dense(
        inputs=input_layer,
        units=units,
        activation=activation,
        weights_initializer=tf.truncated_normal_initializer(0.0, 0.01),
        weights_regularizer=tf.l2_regularizer(0.0005)
    )
    add_variable_summary(layer, 'dense')
    return layer

yolo = tf.pad(images, np.array([[0, 0], [3, 3], [3, 3], [0, 0]]),
name='pad_1')
yolo = convolution_layer(yolo, 64, 7, 2)
yolo = pooling_layer(yolo, [2, 2], 2, 'same')
yolo = convolution_layer(yolo, 192, 3)
yolo = pooling_layer(yolo, 2, 'same')
yolo = convolution_layer(yolo, 128, 1)
yolo = convolution_layer(yolo, 256, 3)
yolo = convolution_layer(yolo, 256, 1)
yolo = convolution_layer(yolo, 512, 3)
yolo = pooling_layer(yolo, 2, 'same')
yolo = convolution_layer(yolo, 256, 1)
yolo = convolution_layer(yolo, 512, 3)
yolo = convolution_layer(yolo, 256, 1)
yolo = convolution_layer(yolo, 512, 3)
yolo = convolution_layer(yolo, 256, 1)
yolo = convolution_layer(yolo, 512, 3)
```

```
yolo = convolution_layer(yolo, 256, 1)
yolo = convolution_layer(yolo, 512, 3)
yolo = convolution_layer(yolo, 512, 1)
yolo = convolution_layer(yolo, 1024, 3)
yolo = pooling_layer(yolo, 2)
yolo = convolution_layer(yolo, 512, 1)
yolo = convolution_layer(yolo, 1024, 3)
yolo = convolution_layer(yolo, 512, 1)
yolo = convolution_layer(yolo, 1024, 3)
yolo = convolution_layer(yolo, 1024, 3)
yolo = tf.pad(yolo, np.array([[0, 0], [1, 1], [1, 1], [0, 0]]))
yolo = convolution_layer(yolo, 1024, 3, 2)
yolo = convolution_layer(yolo, 1024, 3)
yolo = convolution_layer(yolo, 1024, 3)
yolo = tf.transpose(yolo, [0, 3, 1, 2])
yolo = tf.layers.flatten(yolo)
yolo = dense_layer(yolo, 512)
yolo = dense_layer(yolo, 4096)

dropout_bool = tf.placeholder(tf.bool)
yolo = tf.layers.dropout(
    inputs=yolo,
    rate=0.4,
    training=dropout_bool
)
yolo = dense_layer(yolo, output_size, None)
```

EX1010, 128-130.

Although the code snippet uses L2 regularization, Shanmugamani teaches overfitting is addressed by “regularizing using techniques such as dropout and batch normalization,” which are standard TensorFlow functions. EX1010, 77.

Thus, Shanmugamani expressly confirms that the conventional dual head object detection pattern of bounding box regression and class prediction is implemented with hidden layers that include convolution, activation, and batch normalization layers of the type claimed. EX1003, ¶¶348-351.

3. MTC

A POSITA would have combined Chun-Chao-Shanmugamani. Chun discloses a TensorFlow CNN for seat-belt detection, but does not enumerate the standard hidden-layer recipe. Chun states each head uses “two 3×3 convolutions and a 1×1 convolution” for pixel-wise outputs, EX1005, 5, but is silent on the activation / normalization sequencing. Chao likewise refers to neural networks with “one or more hidden layers” for vehicle-object classification. EX1006, ¶¶[0075], [0098]. Shanmugamani provides those missing implementation details by teaching standard TensorFlow object-detection heads built from convolution, activation, and batch-normalization layers. EX1003, ¶352.

A POSITA would have applied those conventional TensorFlow patterns to Chun’s TensorFlow seat-belt head because Shanmugamani expressly teaches how to build and deploy such models for automotive perception and states the reader “will learn how to train deep learning models for computer vision applications and deploy them on multiple platforms.” Shanmugamani p. 6. This is the predictable use of prior art according to their established functions, which is exactly the type of combination *KSR* identifies as obvious. *KSR*, 550 U.S. at 417; EX1003 ¶352.

A POSITA would have expected straightforward success. Chun-Chao-Shanmugamani renders obvious claim 4. EX1003, ¶¶353-356.

C. Claim 5

Chun-Chao-Shanmugamani renders obvious claim 5, depending from claim 1, and Chun-Chao renders claim 1 obvious. EX1003, ¶¶357-358.

1. Chun-Chao

Chun discloses pose estimation heads. Chun teaches keypoint and PAF heads that perform “2D driver/passenger pose estimation.” EX1005, 1. Chun further states that each branch “employs two 3×3 convolutions and a 1×1 convolution to predict a pixel-wise probability distribution.” EX1005, 5. These are TensorFlow implemented pose heads with multiple convolutional layers, but Chun does not spell out the conventional components within those hidden layers. Shanmugamani provides those missing TensorFlow details. EX1003, ¶359.

2. Shanmugamani

Shanmugamani teaches a neural network “is a collection of perceptrons and activation functions,” where “inputs of x are passed through a hidden layer of perceptrons and summed to the output,” and “every layer can have one of the activation functions.” EX1010, 11. Activation functions “make neural nets nonlinear” and are used for gradient adjustment, including sigmoid, tanh, and ReLU. EX1010, 7-9. Shanmugamani teaches batch normalization which “increase[s] the stability and performance of neural network training” by “normaliz[ing] the output from a layer with zero mean and a standard deviation of 1.” EX1010, 13.

Shanmugamani describes pose detection is “the act of finding joint locations in a human,” by detecting “multiple locations such as head, neck, shoulders, ankles, and hands.” EX1010, 117-118.



EX1010, 117.

Shanmugamani describes object localization in the same regression encoder and classification encoder pattern that independently outputs bounding box coordinates and labels. EX1010, 110, 114, 119. Shanmugamani teaches standard TensorFlow retraining on custom datasets including self-driving examples. The example code is as follows:

```
def pooling_layer(input_layer, pool_size=[2, 2], strides=2,
padding='valid'):
    layer = tf.layers.max_pooling2d(
        inputs=input_layer,
        pool_size=pool_size,
        strides=strides,
        padding=padding
    )
    add_variable_summary(layer, 'pooling')
    return layer
```

```
def convolution_layer(input_layer, filters, kernel_size=[3, 3],
padding='valid',
                        activation=tf.nn.leaky_relu):
    layer = tf.layers.conv2d(
        inputs=input_layer,
        filters=filters,
        kernel_size=kernel_size,
        activation=activation,
        padding=padding,
        weights_initializer=tf.truncated_normal_initializer(0.0, 0.01),
        weights_regularizer=tf.l2_regularizer(0.0005)
    )
    add_variable_summary(layer, 'convolution')
    return layer

def dense_layer(input_layer, units, activation=tf.nn.leaky_relu):
    layer = tf.layers.dense(
        inputs=input_layer,
        units=units,
        activation=activation,
        weights_initializer=tf.truncated_normal_initializer(0.0, 0.01),
        weights_regularizer=tf.l2_regularizer(0.0005)
    )
    add_variable_summary(layer, 'dense')
    return layer

yolo = tf.pad(images, np.array([[0, 0], [3, 3], [3, 3], [0, 0]]),
name='pad_1')
yolo = convolution_layer(yolo, 64, 7, 2)
yolo = pooling_layer(yolo, [2, 2], 2, 'same')
yolo = convolution_layer(yolo, 192, 3)
yolo = pooling_layer(yolo, 2, 'same')
yolo = convolution_layer(yolo, 128, 1)
yolo = convolution_layer(yolo, 256, 3)
yolo = convolution_layer(yolo, 256, 1)
yolo = convolution_layer(yolo, 512, 3)
yolo = pooling_layer(yolo, 2, 'same')
yolo = convolution_layer(yolo, 256, 1)
yolo = convolution_layer(yolo, 512, 3)
yolo = convolution_layer(yolo, 256, 1)
yolo = convolution_layer(yolo, 512, 3)
yolo = convolution_layer(yolo, 256, 1)
yolo = convolution_layer(yolo, 512, 3)
```

```
yolo = convolution_layer(yolo, 256, 1)
yolo = convolution_layer(yolo, 512, 3)
yolo = convolution_layer(yolo, 512, 1)
yolo = convolution_layer(yolo, 1024, 3)
yolo = pooling_layer(yolo, 2)
yolo = convolution_layer(yolo, 512, 1)
yolo = convolution_layer(yolo, 1024, 3)
yolo = convolution_layer(yolo, 512, 1)
yolo = convolution_layer(yolo, 1024, 3)
yolo = convolution_layer(yolo, 1024, 3)
yolo = tf.pad(yolo, np.array([[0, 0], [1, 1], [1, 1], [0, 0]]))
yolo = convolution_layer(yolo, 1024, 3, 2)
yolo = convolution_layer(yolo, 1024, 3)
yolo = convolution_layer(yolo, 1024, 3)
yolo = tf.transpose(yolo, [0, 3, 1, 2])
yolo = tf.layers.flatten(yolo)
yolo = dense_layer(yolo, 512)
yolo = dense_layer(yolo, 4096)

dropout_bool = tf.placeholder(tf.bool)
yolo = tf.layers.dropout(
    inputs=yolo,
    rate=0.4,
    training=dropout_bool
)
yolo = dense_layer(yolo, output_size, None)
```

EX1010, 128-130.

In this example, the hidden layers include a convolutional layer, an activation layer (e.g., `tf.nn.leaky_relu`), and a regularization layer (e.g., L2 regularizer), and Shanmugamani explains overfitting may also be addressed using dropout or batch normalization. Shanmugamani further shows the output layer is fed by convolutional features and thus comprises a convolutional layer input. EX1010, 77; EX1003, ¶¶360-365.

3. MTC

A POSITA would have combined Chun-Chao-Shanmugamani for claim 5 for the same structural reason as claim 3, applied here to pose. Chun already discloses a TensorFlow-implemented pose-estimation head using “two 3×3 convolutions and a 1×1 convolution to predict a pixel-wise probability distribution.” EX1005, 5. Chun therefore identifies the task (pose estimation) and framework (TensorFlow) but omits the standard sub-layer details in the hidden layers. Shanmugamani supplies those conventional TensorFlow layer components, teaching that a neural network “is a collection of perceptrons and activation functions” and that each hidden layer uses convolution, batch-normalization, and activation, and that the output head is also convolutional. EX1010, 6-13. Because Chun already uses TensorFlow for this prediction head, a POSITA would simply apply Shanmugamani’s conventional TensorFlow pattern to implement Chun’s pose head. This substitution would have been routine and predictable because Shanmugamani is expressly written as a TensorFlow implementation guide for CNN heads, including vision models for autonomous-vehicle workloads. EX1010, 6. Chun-Chao-Shanmugamani renders obvious claim 5. EX1003, ¶¶366-369.

D. Claims 15-17 and 21-23

Claims 15-17 and 21-23 recite the same features as claims 3-5 as CRM and device rather than method. Claim 15 is the CRM counterpart of claim 3, specifying

bounding-box regression and object-class prediction networks. Claim 16 corresponds to claim 4, reciting that those networks include hidden layers with convolution, batch normalization, and activation. Claim 17 corresponds to claim 5, reciting that the pose-estimation head includes hidden layers with convolution, batch normalization, activation, and a convolutional output. Likewise, claims 21-23 are the device counterparts of claims 3-5. Claim 21 corresponds to claim 3, claim 22 corresponds to claim 4, and claim 23 corresponds to claim 5, but expressed as logic executed by a processor with model parameters stored in memory. EX1003, ¶¶371-372.

The only differences are the statutory form (method vs. CRM vs. device). A POSITA would have understood that the same multi-task CNN disclosed and rendered obvious for claims 3-5 would be implemented either as software instructions on a non-transitory medium or as logic executed in a processor-based device. EX1003, ¶¶373-374.

Chao already discloses implementing distraction classification in software and hardware, including on-device processors, memory, and camera modules. EX1006, ¶¶[0092]-[0093]. Accordingly, for the same reasons set forth for claims 3-5, claims 15-17 and 21-23 would also have been obvious in view of Chun-Chao-Shanmugamani. EX1003, ¶¶375-376.

VII. GROUND 3: ZHENGYANG IN VIEW OF HE RENDERS OBVIOUS CLAIMS 1-6, 9-18, AND 20-24.

Zhengyang in view of He renders independent claims 1, 13 and 20 obvious. Zhengyang discloses the shared-backbone, multi-task CNN architecture for driver-state analysis. He provides the additional pose-estimation head that operates over the same shared feature maps. EX1003, ¶¶377-378. The dependent claims fall with the independents for the same reasons.

A POSITA would have been motivated to combine Zhengyang-He because both references address the same technical problem space of multitask visual human state recognition using a shared-backbone architecture and both provide complementary teachings that a skilled practitioner would have naturally integrated to improve driver state recognition performance. Zhengyang provides the foundation: a unified neural network model with a shared backbone and multiple task heads jointly trained through forward and backward propagation to “simultaneously recognize multiple features of the driver” and thereby “improve the accuracy of driver state recognition.” EX1007, ¶¶[0003], [0005], [0046]. He then teaches precisely the type of additional branch a POSITA would have expected to attach to such a framework. He discloses a pose estimation “keypoint head” comprising convolutional and deconvolutional layers that predicts one hot keypoint masks for each joint type (for example left shoulder and right elbow) and that is trained jointly with detection and segmentation heads while sharing the

same backbone. EX1008, 7-8. The model in He “simultaneously predict[s] boxes, segments, and keypoints” in real time and consumes the same backbone feature maps for all heads. *Id.* A POSITA would have understood that pose estimation directly complements Zhengyang’s distraction-classification tasks because keypoint geometry, such as head tilt, shoulder rotation, and hand position, are highly discriminative features for detecting whether the driver is distracted. Moreover, by the ’243 patent priority date, adding a pose estimation head to a shared backbone multitask model was a routine and well-understood extension that required no change to the underlying training paradigm but merely the definition of another task head with its own labels and loss. Both references employ standard deep learning building blocks including ResNet style backbones, convolutional task heads, cross entropy losses, and gradient based optimization. A POSITA would have had a reasonable expectation of success in integrating He’s keypoint head into Zhengyang’s multitask system and would have expected predictable improvement in driver monitoring performance using ordinary skill rather than inventive ingenuity. EX1003, ¶¶379-385.

A. Independent Claim 1

1. [1Pre]

Zhengyang discloses a method by teaching acquiring driver images, inputting them into the multi-task network, generating feature maps, and outputting distraction-state results. EX1007, ¶¶[0020]-[0023], [0028], [0031], [0046]; EX1003, ¶¶386-387.

2. [1A]

Zhengyang discloses [1A]. Zhengyang describes supervised multi-task learning where annotated driver images are input into a neural network to train parameters. EX1003, ¶¶388-389. Zhengyang explains preprocessed driver images are input to the network “for attribute recognition” to determine whether the driver “is wearing a mask,” “is wearing eyeglasses,” “is smoking,” or “is making a phone call.” EX1007, ¶[0012]. Zhengyang makes clear this same structure is used during training: “before training... it is necessary to label the driver” and “the labeling information and driver images will be used as input to train the model.” EX1007, ¶[0063]; *see also* EX1007, ¶[0025] (“labels the sample image data with multi-task attributes”).

Zhengyang teaches inputting labeled examples into a neural network. EX1003, ¶¶390-393.

3. [1B] and [1C]

Zhengyang discloses both a backbone that extracts shared features from

driver images and task-specific prediction heads fed by those backbone features.

Zhengyang describes a conventional supervised multi-task CNN pipeline in which “the collected RGB image is input into a backbone network to extract encoded features,” EX1007, ¶[0060], and those shared feature maps are then routed to separate attribute-recognition modules. EX1003, ¶¶394-395.

Zhengyang identifies the backbone as SE-ResNet18: “the backbone network uses the SE-ResNet18 network.” EX1007, ¶[0069]. The disclosure confirms how SE-ResNet18 is actually implemented: 112×112 crop, first 7×7→3×3, removal of initial pooling, stride modifications, BN / PReLU / dropout, EX1007, ¶[0100], e.g., standard refinements for cropped in-cabin faces. The 7×7 feature maps are then processed by OSME to extract features for “mask, eyeglasses, smoking, [and] phone-call” recognition. EX1007, ¶¶[0083], [0069]; EX1003, ¶¶396-400.

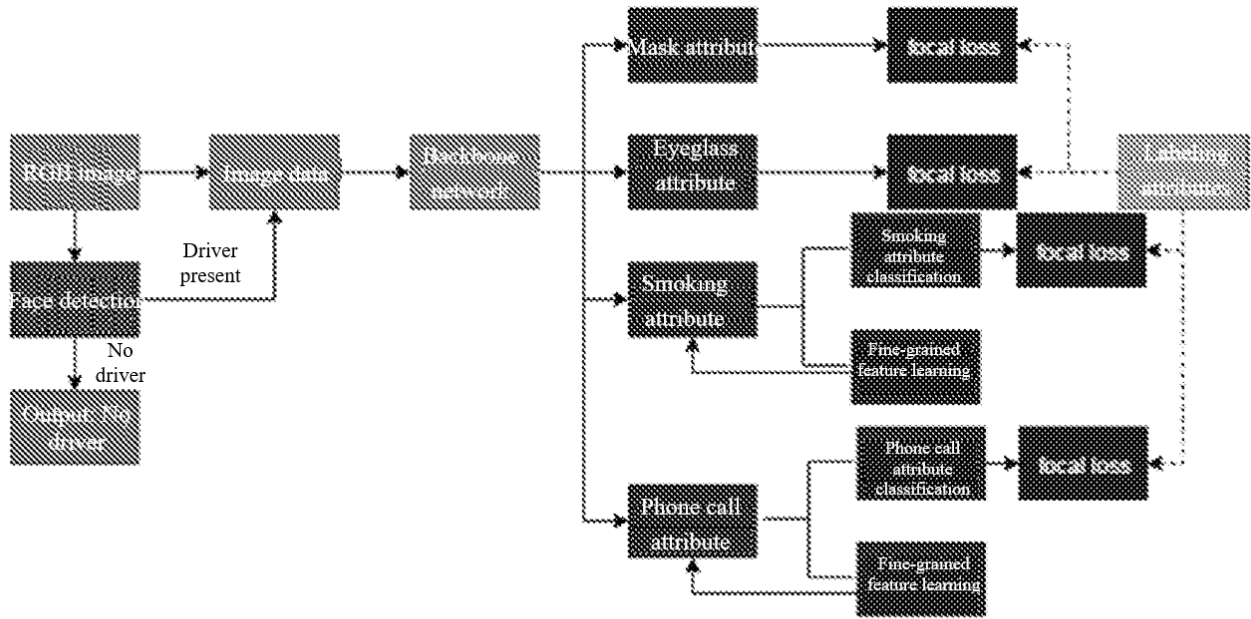


FIG. 3

EX1007, FIG. 3.

Zhengyang states these OSME-derived maps are “input into respective attribute recognition modules” which output “a feature map of each task and an attribute recognition result of each task.” EX1007, ¶¶[0020]-[0021], [0109], FIG. 3. SE-ResNet18 the shared backbone and the attribute-recognition modules are one or more prediction heads coupled to that backbone. EX1003, ¶¶401-404.

4. [1D]

Zhengyang discloses [1D]. Zhengyang discloses prediction heads classifying smoking and phone use. Zhengyang states the multi-task network performs “mask recognition, eyeglasses recognition, smoking recognition, and phone call recognition,” each as a separate downstream module receiving shared backbone

features. EX1007, ¶[0008]; *see also* EX1007, ¶¶[0020], [0094]. The outputs are binary, as Zhengyang specifies “the number of output categories ... is two categories.” EX1007, ¶[0028]; *see also* EX1007, ¶¶[0095], [0098]. The smoking- and phone-call-recognition modules are binary classification heads that determine whether the driver is engaged in a distraction behavior, i.e., analogous to the claimed distraction-classification prediction head. EX1003, ¶¶406-408.

Zhengyang discloses multiple backbone-coupled prediction heads, including smoking-and phone-call-recognition heads that perform distraction-classification. EX1003, ¶408.

5. [1E]

Zhengyang discloses [1E]. Zhengyang discloses an attention-proposal sub-network (APN) that performs object localization, i.e., the claimed object-detection prediction head. Zhengyang states the APN outputs “center point coordinates of the attention area and the edge lengths of the attention area” used for cropping the region of interest. EX1007, ¶¶[0033]-[0034]; *see also* EX1007, ¶[0101]. A POSITA would recognize this as bounding-box prediction serving the same role as region-proposal heads in two-stage detectors. EX1003, ¶¶409-410.

Zhengyang teaches iterative refinement: the cropped region “will be input into the first-level attention model ... APN ... will further crop ... and input ... into the second-level attention model” EX1007, ¶[0102]; *see also* EX1007,

¶[0103]. A POSITA would view this as hierarchical region-refinement consistent with standard detection practice. EX1003, ¶¶411-412.

Although described in conjunction with the attention module, the APN performs a distinct prediction function: producing coordinate-based bounding-box outputs derived from shared backbone features. EX1003, ¶¶412-413. A POSITA would understand the APN to be an object-detection prediction head separate from the distraction-classification heads within the disclosed multi-task architecture. EX1003, ¶413.

6. [1F]

Zhengyang-He renders obvious [1F].

a. Zhengyang

Zhengyang discloses classification and detection heads, but not a pose-estimation head. Adding such a head would have been obvious in view of He, given Zhengyang's shared SE-ResNet18 backbone producing dense spatial maps and multiple task-specific heads consuming those maps, plus attention-guided cropping capturing spatial cues suitable for pose analysis. EX1003, ¶¶414-416.

b. He

He teaches a pose-estimation prediction head, i.e., a keypoint branch predicting joint locations from shared features. He explains "Mask R-CNN... can easily be extended to human pose estimation," modeling each keypoint as a one-hot mask and predicting K masks (e.g., left shoulder, right elbow). EX1008, 7. A

POSITA would view this as a dedicated sub-network outputting spatial heatmaps.

EX1003, ¶417. The keypoint head “consists of eight 3×3 512-d conv layers, followed by a deconv layer and $2\times$ bilinear upscaling,” trained using cross-entropy over an m^2 -way softmax output, with accuracy measured using AP_{kp} . EX1008, 7-8. He confirms this branch operates in parallel with other heads: “We have a unified model that can simultaneously predict boxes, segments, and keypoints” EX1008, 8.

He expressly teaches a pose-estimation prediction head: an independent branch receiving shared convolutional features and producing spatial heatmaps for joints using a dedicated loss, which is readily integrable into Zhengyang’s backbone-plus-heads design. EX1003, ¶¶418-419.

c. MTC

A POSITA would have been motivated to add He’s pose-estimation head to Zhengyang’s multi-task driver-monitoring backbone to improve driver-state assessment. Zhengyang already detects localized cues (smoking / phone use) but seeks to “comprehensively reflect the driver’s driving state and improve the accuracy of driver state recognition.” EX1007, ¶[0046]. He’s keypoint head adds global pose (head/arm/body orientation) and is expressly modular: Mask R-CNN “can simultaneously predict boxes, segments, and keypoints.” EX1008, 8; EX1003, ¶¶420-421.

Both architectures use shared convolutional features feeding modular heads with separate losses. A POSITA would simply attach He's conv/deconv keypoint-regression branch as another prediction head to Zhengyang's backbone and co-train with labeled keypoints, exactly how He describes training end-to-end.

EX1003, ¶¶422-423. This combination is nothing more than the “predictable use of prior art according to their established functions” and thus obvious under *KSR*.

KSR, 550 U.S. at 417. He already shows Mask R-CNN can output boxes, segments, and keypoints in parallel, confirming multi-task heads coexist naturally.

A POSITA therefore would reasonably expect success. EX1003, ¶¶424-428.

7. [1G]

As noted above, Petitioner is not stating [1G] to recite a “joint loss,” and regardless, Zhengyang-He renders this obvious. EX1003, ¶¶429-430

a. **Zhengyang**

Zhengyang teaches standard end-to-end supervised learning in which each head's output is “compared to the labeling results” and the “difference [is] reduced through the corresponding loss function ... to optimize network parameters.”

EX1007, ¶¶[0022]-[0023]. A POSITA would understand this means losses from the heads backpropagate through shared feature maps, necessarily updating backbone weights. Zhengyang confirms “attribute classification losses ... are reduced through forward and back propagation” and that per-task losses are

combined into a total weighted objective. EX1007, ¶¶[0084], [0128]; EX1003, ¶¶431-438.

b. He and MTC

He likewise trains the pose-estimation head “by minimizing cross-entropy loss” over keypoints. EX1008, 7.

MTC: For the reasons discussed for [1F], adding He’s keypoint-loss term into Zhengyang’s already-disclosed multi-loss objective would have been obvious and is a routine multi-task extension: same backbone, same loss-minimization, simply an additional loss head and no architectural change is required. This is the exact “combination of familiar s according to known methods to yield predictable results” that *KSR* deems obvious. *KSR*, 550 U.S. at 416; EX1003, ¶¶439-442.

8. 1[H]

Zhengyang discloses [1H]. After minimizing task-specific losses “through repeated forward and backward propagation,” “the network converges ... and the parameters in the network are the required final results.” EX1007, ¶[0090]. These “parameters” are the trained weights of both the shared backbone and the task-specific heads obtained via loss minimization. Zhengyang states the system “select[s] the network architecture and model with the best attribute analysis results ... deleting the network that does not participate... to reduce the network model size.” EX1007, ¶[0012]. This shows the trained model (including backbone

+ heads) is retained after training.

Zhengyang then teaches storing and executing that trained model: “after the training... the model is deployed in a real-time monitoring system,” where “the pre-trained model is used to extract feature information ... and attribute recognition results are output.” EX1007, ¶¶[0141]-[0144]. Zhengyang confirms the method “is stored in a computer-readable storage medium.” EX1007, ¶[0146]. The stored “program” includes the trained backbone and head parameters needed for inference.

Zhengyang discloses storing, after minimizing the loss, a trained distraction-classification model comprising backbone-prediction-head parameters. EX1003, ¶¶443-449.

B. Claim 2

Zhengyang states that each distraction-classification head (smoking, phone) includes convolutional, pooling, and fully connected layers. EX1003, ¶¶450-451. This follows from: (1) identification of SE-ResNet-18 as the backbone, EX1007, ¶[0100]; (2) statements the attention networks “have the same network structure as the main network model,” EX1007, ¶[0103]; and (3) per-task feature extraction, pooling, and classification. A POSITA would understand each attention head as an independent CNN sub-network trained jointly with the backbone. EX1007, ¶¶[0031]-[0035], [0100]-[0103], [0113]-[0116].

Convolutional-Layer. SE-ResNet-18 is a convolutional architecture with conv/BN/activation blocks and kernel/stride modifications. EX1007, ¶[0100]. The attention heads “have the same network structure,” EX1007, ¶[0103], and perform analysis on cropped images, EX1007, ¶[0035], so a POSITA would understand each head replicates SE-ResNet-style convolution. EX1003, ¶¶452-464.

Pooling-Layer. Zhengyang states “after the four feature maps are pooled, the feature vectors are input into four attribute-recognition modules.” EX1007, ¶[0101]. Pooling occurs within each head’s path generating the vector input to the classifier. While the “first max-pooling layer is deleted,” EX1007, ¶[0100], later pooling / stride-2 / GAP is retained consistent with SE-ResNet-18. EX1003, ¶¶465-470.

Fully Connected-Layer. Zhengyang identifies FC layers in the attention-proposal sub-network (“two fully connected layers in series,” EX1007, ¶¶[0034], [0101], and FC+softmax in the final classifier, EX1007, ¶¶[0035], [0113]-[0116]. EX1003, ¶¶471-476.

Even if alleged Zhengyang does not enumerate each layer type within the heads, a POSITA would still find them obvious based on (1) Zhengyang’s SE-ResNet framework and (2) standard CNN practice. Zhengyang’s SE-ResNet-18 backbone, EX1007, ¶[0100], reflects the conventional convolution→pooling→fully-connected pipeline, and Zhengyang states the

attention networks “have the same network structure.” EX1007, ¶¶[0103].

Implementing the same layer composition in each head would maintain gradient consistency, reduce training complexity, and accords with Zhengyang’s teaching. By the ’243 patent priority date, CNNs for driver-monitoring universally used this sequence: convolution for spatial extraction, pooling for invariance, fully connected for classification. Because the attention heads share the same input modality and feature scale as the SE-ResNet backbone, a POSITA would expect this same sequence to train stably and yield accurate classification outputs. EX1003, ¶¶477-482.

C. Claim 3

Zhengyang discloses this claim. Zhengyang teaches an object-detection prediction head composed of two sub-networks: (1) the APN, which performs bounding-box regression by predicting coordinates of an attention region; and (2) downstream attention networks, which classify the cropped region. EX1007, ¶¶[0031]-[0035], [0101]-[0103], [0113]-[0116]; EX1003, ¶¶483-484.

Bounding-Box Regression Network. The APN “comprises two fully connected layers ... used to output the center-point coordinates ... and the edge lengths of the attention area.” EX1007, ¶[0034]; *see also* EX1007, ¶[0101]. These four outputs (x, y, width, height) are standard bounding-box parameters used in R-CNN/SSD-style regression. Zhengyang explains that these coordinates guide

cropping/enlargement, which are standard post-regression steps. Thus, the APN provides bounding-box localization within Zhengyang's multi-task architecture. EX1003, ¶¶485-488.

Object-Class Prediction Network. The localized region is then processed by attention networks that “re-judge the attributes.” EX1007, ¶[0102]; *see also* EX1007, ¶[0033]. This is categorical classification. Zhengyang states these attention models “have the same network structure as the main network model,” i.e., SE-ResNet-18 with soft-max classification output. EX1007, ¶¶[0103], [0100]. Zhengyang identifies binary class outputs, EX1007, ¶[0028], and soft-max probability computation, EX1007, ¶¶[0113]-[0116], confirming that the attention networks output class probabilities, i.e., object-class predictions. EX1003, ¶¶489-493.

Zhengyang discloses an integrated detection head: the APN predicts where (bounding-box parameters) and the attention networks predict what (class label). EX1003, ¶494.

D. Claim 4

Zhengyang discloses this claim. Zhengyang discloses both the bounding-box regression network and the object-class prediction network are deep neural networks with multiple hidden layers comprising convolution, batch-norm, and activation. Zhengyang defines the model as SE-ResNet-18 and states its sub-

networks share the same structure, including the attention-proposal and attention-classification modules. EX1007, ¶¶[0033]-[0035], [0100]-[0103]. Both networks adopt SE-ResNet residual blocks consisting of repeated convolutional-Batch_normalization-activation layers. EX1003, ¶¶495-500.

SE-ResNet-18 layer template. Zhengyang states “the multi-task network structure used here is SE-ResNet-18 ... BN layer, PReLU, convolution” EX1007, ¶[0100]. This as the SE-ResNet block pattern, convolutional, Batch_Normalization, activation repeated across depth. EX1003, ¶¶501-504.

Bounding-box regression network (APN). Although Zhengyang states the APN “mainly comprises two fully connected layers,” EX1007, ¶[0034], a POSITA would understand those to be *final* regression layers. Consistent with region-proposal practice, earlier feature-processing layers necessarily use SE-ResNet-style convolutional-Batch_normalization-activation blocks. EX1007, ¶¶[0034], [0100]; EX1003, ¶¶505-507.

Object-class prediction network (attention networks). Zhengyang states the attention networks “have the same network structure as the main network model.” EX1007, ¶[0103]. Because the main network is SE-ResNet-18, EX1007, ¶[0100], each attention network likewise repeats convolutional-Batch_normalization-activation layers. EX1007, ¶¶[0100]-[0103]; EX1003, ¶¶508-509.

Thus, Zhengyang’s explicit “BN layer, PReLU, convolution” disclosure,

EX1007, ¶[0100], and its statement that the attention networks share the same structure, EX1007, ¶[0103], would lead a POSITA to understand that both the APN and the attention networks employ repeated convolutional-Batch_normalization-activation hidden layers, consistent with SE-ResNet and standard Faster-R-CNN-style practice. EX1007, ¶¶[0033], [0100]-[0103]; EX1003, ¶¶510-517.

E. Claim 5

Zhengyang-He renders this claim obvious. EX1003, ¶518.

1. Zhengyang

As established in Claim 1, Zhengyang’s multi-task SE-ResNet-18 framework is designed for multiple task heads and is amenable to adding a pose-estimation head. Zhengyang provides the shared backbone, multi-head context, and explicit layer template of “BN layer, PReLU, convolution.” EX1007, ¶[0100]; EX1003, ¶¶519-520.

2. He

Even if Zhengyang is alleged deficient, He provides a concrete pose-estimation head that operates as a deep convolutional sub-network with multiple hidden layers and a final convolutional output that produces keypoint heatmaps. EX1008, 7-8; EX1003, ¶¶521-525.

Hidden-layer composition in a pose-estimation head. He teaches a keypoint head that “consists of a stack of eight 3×3 512-d convolution layers.” EX1008, 8. That is a plurality of hidden layers. In SE-ResNet based heads, each convolution layer is paired with a batch-normalization layer and an activation layer.

MTC: Zhengyang states SE-ResNet-18 uses batch-normalization, PReLU activation, and convolution. EX1007, ¶[0100]. Zhengyang also states task heads “have the same network structure as the main network model.” EX1007, ¶[0103]. Therefore, a POSITA would implement each hidden layer in the added pose head as a convolution layer, a batch-normalization layer, and an activation layer in order to maintain consistency with the SE-ResNet-18 template and to maintain stable feature scaling across all heads. EX1007, ¶¶[0100], [0103]; EX1003, ¶526.

Output layer of the pose-estimation head. He teaches that the keypoint head produces K keypoint masks and details a final convolutional stage that generates the heatmap outputs, with upsampling via deconvolution and bilinear scaling to arrive at the target resolution. EX1008, 7-8. A POSITA would recognize this as an output layer that comprises a convolutional layer, satisfying the claim’s output-layer requirement. EX1003, ¶¶527-529.

MTC: A POSITA would attach He’s keypoint head as an additional branch to Zhengyang’s shared SE-ResNet-18 feature maps and implement the hidden layers using the same convolution-BN-activation block pattern already specified for

Zhengyang's network. EX1007, ¶[0100]; EX1008, 8. **Further**, Zhengyang seeks to “comprehensively reflect the driver's driving state and improve the accuracy of driver state recognition.” EX1007, ¶[0046]. Adding a pose-estimation head yields keypoint heatmaps for head, shoulders, arms, and related joints, which directly improves assessment of posture and distraction while coexisting with other task heads, as demonstrated in multi-task frameworks. EX1008, 8. **Finally**, noth references use convolutional backbones with task-specific heads, and the layer types are standard modules. He reports successful joint operation of detection, segmentation, and keypoints in one model, indicating that adding a pose branch does not disrupt other tasks. *Id.* Given Zhengyang's explicit “BN layer, PReLU, convolution” template and shared-structure statement, a POSITA would expect straightforward integration and stable training. EX1007, ¶¶[0100], [0103]; EX1003, ¶¶530-534.

F. Claim 6

Petitioner is not reading Claim 6 to recite a dedicated “distraction-classification prediction head,” and regardless, Zhengyang-He renders obvious claim 6.

Zhengyang teaches inputting driver-facing images into a trained multi-task SE-ResNet-18 network and outputting binary attribute results including whether there is smoking and whether a phone call is being made. EX1007, ¶¶[0020]-

[0023], [0028], [0031], [0046], [0061]-[0062], [0100]-[0103], [0113]-[0116].

Zhengyang states each head produces “two categories” (e.g., smoking vs. not smoking) using a soft-max output. EX1007, ¶¶[0028], [0113]-[0116]. These binary outputs are the claimed distraction-classification “tag.” And even if it were argued that Zhengyang does not expressly disclose receiving a “tag,” it would have been obvious in view of He.

He confirms that this is the standard output produced by each CNN classification branch: He discloses heads that output both “class label” and associated offsets, selected via soft-max. EX1008, 3, 8. He explains that the model uses the mask associated with the predicted class selected by the classification branch. EX1008, 8. This is the conventional sequence of a convolutional head followed by a fully connected layer and soft-max that yields a discrete class label. That mechanism corresponds to the “receiving a distraction classification tag” step, because both produce categorical identifiers derived from image data.

MTC. He’s explicit soft-max output therefore confirms the label-generation step implicit in Zhengyang’s attribute-recognition results. EX1003, ¶¶535-546. A POSITA would integrate He’s explicit soft-max class-label output into Zhengyang’s attention-head architecture because both references use the same shared-backbone, multi-head CNN framework with convolution, BN, activation, and fully connected layers. EX1007, ¶¶[0100]-[0103]; EX1008, 3. Zhengyang

seeks to “comprehensively reflect the driver’s driving state and improve the accuracy of driver state recognition.” EX1007, ¶[0046]. Generating categorical labels (e.g., smoking vs. not-smoking) directly advances that objective, and He confirms such class-label heads are standard in modern CNNs and operate in real-time within multi-head models. EX1008, 3, 8. A POSITA therefore would have had a reasonable expectation of success. EX1003, ¶¶547-553.

G. Claim 9

Zhengyang-He renders claim obvious 9. EX1003, ¶554

1. Zhengyang

Zhengyang teaches a multi-task CNN that classifies driver images into categorical distraction states, such as smoking or phone use, to improve driving safety. EX1007, ¶¶[0002], [0021]. These outputs are binary yes or no for each behavior, which are categorical labels derived from the feature vector of each image, consistent with standard soft max classification. EX1007, ¶[0028]. Smoking and phone use correspond to distracted states, and the absence of those behaviors corresponds to undistracted. Zhengyang identifies smoking recognition and phone call recognition heads, which map directly to the claimed mobile usage and other distracted categories. EX1007, ¶[0008]; EX1003, ¶¶555-559.

2. He

He confirms that CNN classification heads produce class labels. Mask R CNN outputs a class label selected by the classification branch. EX1008, 3, 8.

3. MTC

A POSITA would therefore incorporate He's explicit labeling structure into Zhengyang's prediction heads to formalize the categorical outputs already described in Zhengyang. This is obvious because both references use shared backbones and task specific heads that feed fully connected soft max decision layers. EX1003, ¶560. It would have been straightforward to combine these teachings because He's class label format merely clarifies the binary outputs generated by Zhengyang's heads, and directly supports Zhengyang's stated goal to improve driving safety by identifying distraction states. EX1007, ¶[0002]. A POSITA would reasonably expect success because by the priority date, CNN driver monitoring systems were already labeling driver states such as attentive, phone use, or other distraction. EX1003, ¶¶561-565.

H. Claim 10

Zhengyang attention proposal sub network discloses claim 10. The APN performs object localization by outputting spatial parameters indicating where a target object appears in the driver image. Zhengyang states that it comprises two fully connected layers that output the center point coordinates and edge lengths of the attention area for cropping and enlargement of the original input image. EX1007, ¶[0034]. The "input image" is represented as a numerical feature vector produced by the convolutional backbone and that the APN outputs the coordinates

that identify the location of the object within that vector.

Zhengyang further describes iterative refinement in which the APN's predicted region is cropped and reanalyzed for improved localization accuracy. EX1007, ¶¶[0101]-[0102]. This is the standard region proposal and bounding box refinement loop used in conventional object detection.

Accordingly, the APN detects the location of an object such as a phone or cigarette within the input feature vector. EX1003, ¶¶566-572.

I. Claim 11

Zhengyang-He renders obvious claim 11. EX1003, ¶573

Zhengyang converts each driver image into a numerical feature representation through convolutional processing, which is the input vector on which each head operates. EX1003, ¶574.

He teaches a modular pose estimation head receives shared convolutional features and outputs keypoint heatmaps representing the orientation and location of human body parts. He explains the model is extended to pose estimation and describes a keypoint head with stacked convolutional layers followed by deconvolution and bilinear upscaling to produce spatial heatmaps. EX1008, 7-8. These heatmaps represent the orientation of body parts derived directly from the input vector. EX1003, ¶¶575-577.

MTC: Zhengyang already generates localized attention regions focused on the driver's head, hands, and face, which could be used as inputs to a pose estimation head of the type disclosed by He. Integrating this head into Zhengyang's multi task framework would require minimal modification because both references use shared backbones and modular heads. EX1003, ¶¶578-580.

A POSITA would be motivated to add He's pose estimation head to improve Zhengyang's ability to assess distraction, since orientation cues such as head turn or arm position are relevant to attentiveness. A POSITA would expect success due to the architectural compatibility, and the combined system would detect orientation of body parts in the input vector as required by claim 11. EX1003, ¶¶581-585.

J. Claim 12

Zhengyang discloses claim 12. EX1003, ¶586. As discussed for claim 1, Zhengyang teaches storing a trained distraction-classification model comprising backbone and head parameters for inference. Storing that model in the dash mounted camera is a routine implementation. EX1003, ¶¶587-588.

Zhengyang discloses an in-vehicle camera device with onboard processing and memory that acquires the face image and inputs it to the network, confirming a camera module containing sensor, processor, and memory. EX1007, ¶¶[0014], [0045]. Zhengyang also prunes the model to reduce size and forward transmission

time, which indicates local inference, and therefore storage within device memory. EX1007, ¶¶[0038]-[0039]; EX1003, ¶¶589-591. Based on these teachings, Zhengyang's trained model is stored within the dash-mounted camera to minimize latency, reduce wiring complexity, and enable immediate driver-state detection.

K. Claims 13-18 and 20-24

Zhengyang itself discloses that the same multi-task driver state recognition method is embodied as a CRM storing executable instructions, and as a processor-and-memory-based device executing those instructions. EX1007, ¶¶[0044]-[0045], [0137]-[0142], [0143]-[0147]. Thus, for the same reasons discussed above for claims 1-7 and 9-12, a POSITA would have found storing program instructions on a medium (claims 13-18) or executing those same instructions on a processor in a device (claims 20-24) obvious in view of Zhengyang-He. Accordingly, claims 13-18 and 20-24 are obvious for the same reasons as claims 1-7 and 9-12. EX1003, ¶¶592-600.

VIII. GROUND 4: ZHENGYANG IN VIEW OF HE AND FURTHER IN VIEW OF CHAO RENDERS OBVIOUS CLAIMS 7-8, 19, 25-33.

A. Claims 7, 8, 19 and 25

Zhengyang in view of He and further in view of Chao renders obvious claims 7, 8, 19 and 25. Claims 19 and 25 merely restate the same functional recitations of claims 7 and 8, but in CRM and device form. EX1003, ¶¶601-603

1. Zhengyang-He

A POSITA would understand that Zhengyang-He discloses, teach or suggest claims 7, 8, 19 and 25.

Zhengyang produces a distraction-classification tag (for example “smoking” or “phone call”) and teaches these categorical outputs are generated in an intelligent vehicle environment “so as to improve driving safety.” EX1007, ¶[0002]. Zhengyang explicitly identifies the distraction states “whether there is smoking, and whether a phone call is being made,” EX1007, ¶¶[0021], [0028], and describes these outputs “reflect the driver’s driving state and improve the accuracy of driver state recognition,” EX1007, ¶[0046]. A POSITA would not interpret these categorical tags as inert values. In a safety motivated IVMS used “to improve driving safety,” such categorical tags function as decision tokens that drive the next operation in the pipeline. EX1003, ¶¶604-606.

Zhengyang also discloses an integrated in vehicle camera device with a processor, memory, and “an output apparatus with a display and speaker.” EX1007, ¶[0152]; *see also* EX1007, ¶¶[0149]-[0153]. This configuration is a driver monitoring module that performs inference and has the output pathways needed to consume and act on the distraction tag. EX1003, ¶¶607-609.

He independently reinforces that the predicted class is the control variable that determines what happens next. He states that the network outputs “a class

label and a bounding box offset” and that the system “only use[s] the k-th mask where k is the predicted class.” EX1008, 3, 8. This is standard gating logic. Under the breadth of claim 7, this gating behavior itself is an action based on the distraction-classification tag.

For that reason, Zhengyang-He renders obvious claim 7. EX1003, ¶¶610-615.

If it is alleged claim 7 recites an outward user facing action such as visual, audible, or network transmission output (i.e., claim 8), that implementation would have been obvious further in view of Chao, which explicitly teaches the exact downstream actuation contemplated by claims 7, 8, 19, and 25.

2. Chao

Chao teaches a driver-monitoring CNN that outputs a class label and then uses that determined state to “trigger[] vehicular actions based on the determined state.” EX1006, ¶¶[0022], [0068], [0095], [0129]. Those actions include notifications “via the UIs,” which “may include a visual notification, a sound notification, or a vibration notification.” EX1006, ¶¶[0095], [0117], [0168]. A visual notification is updating a display, a sound notification is an audible alarm, and that the determined state is delivered to a separate processor. EX1006, ¶¶[0032], [0075], [0078].

Accordingly, Chao provides the mechanism that maps the classification output to display updates, alarms, or remote transmission, which are the actions recited in claim 8. EX1003, ¶¶616-620.

3. MTC

A POSITA would view Zhengyang-He-Chao as three adjacent stages of the same pipeline: Zhengyang generates the categorical distraction output, He shows that a class ID is the control signal that determines what operation executes next, and Chao consumes the class/state to drive HMI or vehicle actuation. The integration is mechanically trivial: the discrete class value or softmax output produced by the Zhengyang task head would be passed directly into the notification or actuation module described in Chao, with no new network layers, no feature-map changes, and no data-type adaptation. All three references assume the same modality (driver-facing images), the same architecture family (multi-head CNN), and the same deployment surface (in-vehicle runtime with display/speaker/control channels). The technical alignment across the three references means a POSITA would expect success on first implementation, because this is the standard deployment pattern: perception, classification, then actuation *KSR*, 550 U.S. at 416-417. Claims 7, 8, 19 and 25 therefore add no features beyond routine label-based control that a POSITA would have expected to apply. EX1003, ¶¶621-625.

B. Claims 26-33

Claims 26-33 recite no new features, rather they merely restate the same limitations of claims 1-12 in the context of a dash-mounted in-vehicle inference device. Independent claims 26 and 30 are the method/device counterparts, and claims 27-29 and 31-33 track the same refinements that appear in claims 9-11.

Claim 26 mirrors the method limitations of claims 1, 6-8, and 12, but placed in a dash-mounted camera device. For the same reasons already discussed for those claims, storing the trained model in the dash-mounted unit, capturing in-cabin images, inputting the images into the model, receiving the distraction tag, and performing an action based on the tag would have been obvious in view of Zhengyang-He-Chao. Zhengyang describes the same in-vehicle capture→inference pipeline, EX1007, ¶¶[0060], [0109], [0145]; He confirms multi-head architectures trained end-to-end; and Chao expressly teaches triggering actions based on driver-state classification, including “visual,” “sound,” or remote notifications. EX1006, ¶¶[0095], [0168]; EX1003 ¶¶626-631.

Claims 27-29 add no features beyond claim 26 and correspond to the same refinements addressed in claims 9-11 (e.g., classification labels, bounding-box location, and pose orientation) using image data instead of an input vector, which would have been obvious for the same reasons in view of Zhengyang-He-Chao as the input vector is generated based on image data of drivers. EX1003, ¶632.

Claim 30 simply recasts claim 26 as a processor-and-memory device.

Zhengyang already discloses the same trained backbone+heads model embodied as executable instructions stored in memory and executed by a processor within an in-vehicle unit. EX1007, ¶¶[0044]-[0045], [0137]-[0146]. Claims 31-33 add the same dependent refinements as claims 27-29, again with no new substance. EX1003, ¶¶633-635.

Accordingly, claims 26-33 merely restate the same features in CRM/device form. A POSITA would have found each obvious for the reasons already articulated in view of Zhengyang-He-Chao.

IX. NO OBJECTIVE INDICIA OF PATENTABILITY

Petitioner is unaware of any objective indicia of non-obviousness. If alleged in the POPR, Petitioner requests a response. EX1003, ¶¶636-639.

X. MANDATORY NOTICES

A. Real Party in Interest

Petitioner Samsara Inc. (“Samsara”) is the only real party-in-interest.

B. Related Matters

Motive asserted the ’243 patent in the Northern District of California (Case No. 3-24-cv-00902) in an amended complaint filed on July 9, 2025. EX1037. In the same amended complaint, Motive also asserted U.S. Patent No. 12,136,276 (the “’276 patent”). The ’276 patent is a child of the ’580 patent. An IPR trial (IPR2025-00574) was instituted against the ’580 patent on August 27, 2025. An IPR was filed against the ’276 patent on October 17, 2025 (IPR2026-00034).

A. Lead and Backup Counsel

Lead Counsel	Back-up Counsel
Jason D. Eisenberg (Reg. No. 43,447) jasone-PTAB@sternekessler.com STERNE, KESSLER, GOLDSTEIN & FOX PLLC 1101 K Street, NW, 10th Floor Washington, DC 20005 202.371.2600 (reception) 202.371.2540 (facsimile)	Lestin Kenton (Reg. No. 72,314) lkenton-PTAB@sternekessler.com Bryn Mendenhall (Reg. No. 82,263) bmendenhall-PTAB@sternekessler.com STERNE, KESSLER, GOLDSTEIN & FOX PLLC 1101 K Street, NW, 10th Floor Washington, DC 20005 202.371.2600 (reception) 202.371.2540 (facsimile)

Please address all correspondence to lead and backup counsel at the email addresses shown above and PTAB@sternekessler.com. Samsara consents to electronic service by email.

XI. GROUNDS FOR STANDING

Samsara certifies the '243 patent is available for IPR and Samsara is not barred or estopped from requesting IPR of the '243 patent challenging the patent claims on the Grounds identified in this Petition.

XII. CONCLUSION

The Board should institute *inter partes* review and cancel all challenged claims.

Respectfully submitted,

STERNE, KESSLER, GOLDSTEIN & FOX PLLC

/Jason D. Eisenberg/

Jason D. Eisenberg, Reg. No. 43,447
Lestin Kenton, Reg. No. 72,314
Bryn Mendenhall, Reg. No. 82,263
Counsel for Petitioner Samsara Inc.

Date: November 14, 2025

1101 K Street NW, 10th Floor
Washington, DC 20005
(202) 371-2600

Claims Appendix – U.S. Patent No. 12,062,243

Claim 1

Element	
[1P]	A method comprising
[1A]	inputting labeled examples into a neural network, the neural network comprising:
[1B]	a backbone network, and
[1C]	one or more prediction heads coupled to the backbone network, the one or more prediction heads comprising one or more of
[1D]	a distraction classification prediction head
[1E]	object detection prediction head
[1F]	and a pose estimation prediction head
[1G]	minimizing a loss based on outputs of the one or more prediction heads causing a change in parameters of the backbone network; and
[1.H]	storing a distraction classification model after minimizing the loss, the distraction classification model comprising the parameters of the backbone network and parameters of at least one of the one or more prediction heads.

Claim 2

Element	
[2]	The method of claim 1, wherein the distraction classification prediction head includes a convolutional layer, pooling layer, and fully connected layer.

Claim 3

Element	
[3]	The method of claim 1, wherein the object detection prediction head comprises a bounding box regression network and an object class prediction network, wherein the bounding box regression network outputs coordinates of a bounding box enclosing a detected object and the object class prediction network outputs a class corresponding to the detected object.

Claim 4

Element	
[4]	The method of claim 3, wherein each of the bounding box regression network and the object class prediction network comprise deep neural networks having a plurality of hidden layers, each hidden layer in the plurality of hidden layers comprising a convolutional layer, a batch normalization layer, and a batch activation layer.

Claim 5

Element	
[5]	The method of claim 1, wherein the pose estimation prediction head includes a plurality of hidden layers and an output layer, each of the plurality of hidden layers comprising a convolutional layer, a batch normalization layer, and an activation layer, and the output layer comprising a convolutional layer.

Claim 6

Element	
[6]	The method of claim 1, further comprising inputting image data into the distraction classification model and receiving a distraction classification tag from the distraction classification model based on the image data.

Claim 7

Element	
[7]	The method of claim 6, further comprising performing at least one action based on the distraction classification tag.

Claim 8

Element	
[8]	The method of claim 7, the at least one action comprising one or more of updating a display of a computing device, playing an audible alarm, transmitting or a prediction to a remote server.

Claim 9

Element	
[9]	The method of claim 1, wherein the distraction classification prediction head classifies an input vector with a label, the label comprising one or more of an undistracted, mobile usage, or other distracted label.

Claim 10

Element	
[10]	The method of claim 1, wherein the object detection prediction head detects a location of an object within an input vector.

Claim 11

Element	
[11]	The method of claim 1, wherein the pose estimation prediction head detects an orientation of one or more body parts of a user depicted in an input vector.

Claim 12

Element	
[12]	The method of claim 1, wherein storing a distraction classification model comprises storing the distraction classification model in a dash-mounted camera device.

Claim 13

Element	
[13P]	A non-transitory computer-readable storage medium for tangibly storing computer program instructions capable of being executed by a computer processor, the computer program instructions defining steps of:
[13A]	inputting labeled examples into a neural network, the neural network comprising:
[13B]	a backbone network, and one or more prediction heads,
[13C]	the one or more prediction heads selected from a group consisting of a distraction classification prediction head, object detection prediction head, and a pose estimation prediction head;
[13D]	a distraction classification prediction head,
[13E]	object detection prediction head,
[13F]	and a pose estimation prediction head;
[13G]	minimizing a loss based on outputs of the one or more prediction heads causing a change in parameters of the backbone network; and
[13H]	storing a distraction classification model after minimizing the loss, the distraction classification model comprising the parameters of the backbone network and parameters of at least one of the one or more prediction heads.

Claim 14

Element	
[14]	The non-transitory computer-readable storage medium of claim 13, wherein the distraction classification prediction head includes a convolutional layer, pooling layer, and fully connected layer.

Claim 15

Element	
[15]	The non-transitory computer-readable storage medium of claim 13, wherein the object detection prediction head comprises a bounding box regression network and an object class prediction network, wherein the bounding box regression network outputs coordinates of a bounding box enclosing a detected object and the object class prediction network outputs a class corresponding to the detected object.

Claim 16

Element	
[16]	The non-transitory computer-readable storage medium of claim 15, wherein each of the bounding box regression network and the object class prediction network comprise deep neural networks having a plurality of hidden layers, each hidden layer in the plurality of hidden layers comprising a convolutional layer, a batch normalization layer, and a batch activation layer.

Claim 17

Element	
[17]	The non-transitory computer-readable storage medium of claim 13, wherein the pose estimation prediction head includes a plurality of hidden layers and an output layer, each of the plurality of hidden layers comprising a convolutional layer, a batch normalization layer, and an activation layer, and the output layer comprising a convolutional layer.

Claim 18

Element	
[18]	The non-transitory computer-readable storage medium of claim 13, the steps further comprising inputting image data into the distraction classification model and receiving a distraction classification tag from the distraction classification model based on the image data.

Claim 19

Element	
[19]	The non-transitory computer-readable storage medium of claim 18, the steps further comprising performing at least one action based on the distraction classification tag.

Claim 20

Element	
[20P]	A device comprising:
[20A]	a processor; and
[20B]	a storage medium for tangibly storing thereon program logic for execution by the processor, the program logic comprising:
[20C]	logic, executed by the processor, for inputting labeled examples into a neural network, the neural network comprising: a backbone network, and one or more prediction heads, the one or more prediction heads selected from a group consisting of a distraction classification prediction head, object detection prediction head, and a pose estimation prediction head,
[20D]	logic, executed by the processor, for minimizing a loss based on outputs of the one or more prediction heads causing a change in parameters of the backbone network, and
[20E]	logic, executed by the processor, for storing a distraction classification model after minimizing the loss, the distraction classification model comprising the parameters of the backbone network and parameters of at least one of the one or more prediction heads.

Claim 21

Element	
[21]	The device of claim 20, wherein the object detection prediction head comprises a bounding box regression network and an object class prediction network, wherein the bounding box regression network outputs coordinates of a bounding box enclosing a detected object and the object class prediction network outputs a class corresponding to the detected object.

Claim 22

Element	
[22]	The device of claim 21, wherein each of the bounding box regression network and the object class prediction network comprise deep neural networks having a plurality of hidden layers, each hidden layer in the plurality of hidden layers comprising a convolutional layer, a batch normalization layer, and a batch activation layer.

Claim 23

Element	
[23]	The device of claim 20, wherein the pose estimation prediction head includes a plurality of hidden layers and an output layer, each of the plurality of hidden layers comprising a convolutional layer, a batch normalization layer, and an activation layer, and the output layer comprising a convolutional layer.

Claim 24

Element	
[24]	The device of claim 20, the program logic further comprising logic, executed by the processor, for comprising inputting image data into the distraction classification model and receiving a distraction classification tag from the distraction classification model based on the image data.

Claim 25

Element	
[25]	The device of claim 24, the program logic further comprising logic, executed by the processor, for performing at least one action based on the distraction classification tag.

Claim 26

Element	
[26.P]	A method comprising:
[26A]	storing a distraction classification model in a dash-mounted camera device, the distraction classification model comprising:
[26B]	a backbone network and one or more prediction heads coupled to the backbone network, the one or more prediction heads comprising one or more of a distraction classification prediction head, object detection prediction head, and a pose estimation prediction head, wherein the distraction classification model comprises a model trained by inputting labeled examples into the distraction classification model and minimizing a loss based on outputs of the one or more prediction heads causing a change in parameters of the backbone network;
[26C]	capturing an image of an interior of a vehicle using a sensor in the dash-mounted camera device;
[26D]	inputting the image into the distraction classification model;
[26E]	receiving a distraction classification tag output by the distraction classification model; and
[26F]	performing at least one action by the dash-mounted camera device based on the distraction classification tag, the at least one action comprising one or more of updating a display of a computing device, playing an audible alarm, transmitting or a prediction to a remote server.

Claim 27

Element	
[27]	The method of claim 26, wherein the distraction classification prediction head classifies the image with a label, the label comprising one or more of an undistracted, mobile usage, or other distracted label.

Claim 28

Element	
[28]	The method of claim 26, wherein the object detection prediction head detects a location of an object within the image.

Claim 29

Element	
[29]	The method of claim 26, wherein the pose estimation prediction head detects an orientation of one or more body parts of a user depicted in the image.

Claim 30

Element	
[30P]	A dash-mounted camera device comprising:
[30A]	a memory configured to:
[30B]	store a distraction classification model, the distraction classification model comprising:
[30C]	a backbone network and one or more prediction heads coupled to the backbone network, the one or more prediction heads comprising one or more of a distraction classification prediction head, object detection prediction head, and a pose estimation prediction head, wherein the distraction classification model comprises a model trained by inputting labeled examples into the distraction classification model and minimizing a loss based on outputs of the one or more prediction heads causing a change in parameters of the backbone network;
[30D]	a sensor oriented toward an interior of a vehicle; and
[30E]	a processor configured to:
[30F]	capture an image of an interior of a vehicle;
[30G]	input the image into the distraction classification model;
[30H]	receive a distraction classification tag output by the distraction classification model; and

[30I]	perform at least one action by the dash-mounted camera device based on the distraction classification tag, the at least one action comprising one or more of updating a display of a computing device, playing an audible alarm, transmitting or a prediction to a remote server.
-------	---

Claim 31

Element	
[31]	The dash-mounted camera device of claim 30, wherein the distraction classification prediction head classifies the image with a label, the label comprising one or more of an undistracted, mobile usage, or other distracted label.

Claim 32

Element	
[32]	The dash-mounted camera device of claim 30, wherein the object detection prediction head detects a location of an object within the image.

Claim 33

Element	
[33]	The dash-mounted camera device of claim 30, wherein the pose estimation prediction head detects an orientation of one or more body parts of a user depicted in the image.

**CERTIFICATE OF COMPLIANCE WITH TYPE-VOLUME
LIMITATION, TYPEFACE REQUIREMENTS, AND TYPE STYLE
REQUIREMENTS**

1. This Petition complies with the type-volume limitation of 14,000 words, comprising 13,639 words, excluding the parts exempted by 37 C.F.R. § 42.24(a).

2. This Petition complies with the general format requirements of 37 C.F.R. § 42.6(a) and has been prepared using Microsoft® Word in 14-point Times New Roman.

Respectfully submitted,

STERNE, KESSLER, GOLDSTEIN & FOX PLLC

/Jason D. Eisenberg/

Jason D. Eisenberg
Registration No. 43,447
Attorney for Petitioner Samsara Inc.

Date: November 14, 2025

1101 K Street NW, 10th Floor
Washington, DC 20005
(202) 371-2600

CERTIFICATION OF SERVICE (37 C.F.R. §§ 42.6(e), 42.105(a))

The undersigned hereby certifies that the above-captioned **PETITION FOR**
***INTER PARTES* REVIEW OF U.S. PATENT NO. 12,062,243** and all
supporting exhibits were served via FedEx Express® on November 14, 2025, in
their entirety on the following:

GREENBERG TRAURIG (NY)
One Vanderbilt Avenue
New York, NY 10017
Correspondence Address for U.S. Patent No. 12,062,243

Respectfully submitted,

STERNE, KESSLER, GOLDSTEIN & FOX PLLC

/Jason D. Eisenberg/

Jason D. Eisenberg
Registration No. 43,447
Attorney for Petitioner Samsara Inc.

Date: November 14, 2025

1101 K Street NW, 10th Floor
Washington, DC 20005
(202) 371-2600