Mitigating Reverse Engineering Attacks on Local Feature Descriptors

Deeksha Dangwal[1]
deeksha@cs.ucsb.edu
Vincent T. Lee[2]
vtlee@fb.com
Hyo Jin Kim[2]
hyojinkim@fb.com
Tianwei Shen[2]
tianweishen@fb.com
Meghan Cowan[2]
meghancowan@fb.com
Rajvi Shah[2]
rajvishah@fb.com
Caroline Trippel[3]
trippel@stanford.edu
Brandon Reagen[4]
bjr5@nyu.edu
Timothy Sherwood[1]
sherwood@cs.ucsb.edu
Vasileios Balntas[2]
vassileios@fb.com
Armin Alaghi[2]
alaghi@fb.com
Eddy Ilg[2]
eddyilg@fb.com

[1] University of California, Santa Barbara
Santa Barbara, USA
[2] Facebook Reality Labs Research
Redmond, USA
[3] Stanford University
Stanford, USA
[4] New York University
New York, USA

Abstract

As autonomous driving and augmented reality evolve a practical concern is data privacy, notably when these applications rely on user image-based localization. The widely adopted technology uses local feature descriptors derived from the images. While it was long thought that they could not be reverted back, recent work has demonstrated that under certain conditions reverse engineering attacks are possible and allow an adversary to reconstruct RGB user images. This poses a potential risk to user privacy.

We take this further and model potential adversaries using a privacy threat model. We show a reverse engineering attack on sparse feature maps under controlled conditions and analyze the vulnerability of popular descriptors including FREAK, SIFT and SOSNet. Finally, we evaluate potential mitigation techniques that select a subset of descriptors to carefully balance privacy reconstruction risk. While preserving image matching accuracy, our results show that similar accuracy can be obtained when revealing less information.

1 Introduction

Privacy and security of user data has quickly become a concern and an important design consideration when engineering autonomous driving and augmented reality systems. These systems require always-on information capture in order to support machine perception stacks and rely directly or indirectly on the data that originates from the user's device, i.e., RGB, inertial, depth, and other sensor values. Data assets are rich in private information, but due to the compute power limitations on the device, they must be sent to a service provider to enable services such as localization, and virtual content overlay. As a result, there is understandable concern that any data assets shared with a cloud service provider, no matter how well-trusted, can potentially be abused [12]. To enable augmented reality in practice, beyond the application functionality, privacy-preserving techniques are thus an important consideration.

We focus on localization as a fundamental component of augmented reality. Localization relies on visual data assets to make a prediction of the pose of the user; in particular, most established algorithms rely on local feature descriptors. Since these descriptors contain only derived information, they were long thought to be secure. Unfortunately, recent literature shows that descriptors can be reverse engineered surprisingly well (Figure 1).

For feature descriptors, a reverse engineering attack [6] attempts to reconstruct the original RGB image that was used to derive the feature descriptors. The fidelity to which the original RGB image can be reconstructed is an indication of the severity of the potential risk to privacy. Prior work [19, 23, 34, 49] has shown that feature descriptors are potentially susceptible to such an attack under a range of conditions and configurations. However, there is limited work on quantitatively analyzing privacy implications as well as evaluating potential defenses against such reverse engineering attacks, which our work will explore. To understand the privacy implications, we first utilize a privacy threat model [17] to determine what assets are available to a descriptor reverse-engineering attack and evaluate how information might be leaking. We then propose two mitigation techniques inspired by current best practices in privacy and security [33]: (1) reducing the number of features shared and (2) selective suppression of features around potentially sensitive objects. We show that these techniques can mitigate the potency of reverse engineering attacks on feature descriptors to improve protections on user data. In summary, we make the following contributions:

1. We present a privacy threat model for a reverse engineering attack to narrow down the privacy-critical information and scope the setup for a practical attack.
2. We demonstrate a reverse engineering attack to reconstruct RGB images from sparse feature descriptors such as FREAK [20], SIFT [3] and SOSNet [51], and quantitatively analyze the privacy implications. In contrast to previous work [49, 34], our approach does not take additional information such as sparse RGB, depth, orientation, or scale as input.
3. We present two mitigation techniques to improve local feature descriptor privacy by reducing the number of keypoints shared for localization. We show that there is a trade-off between enhanced privacy (less fidelity of reconstruction) and the utility (localization accuracy). We also show that the choice of shared keypoints matters for privacy.

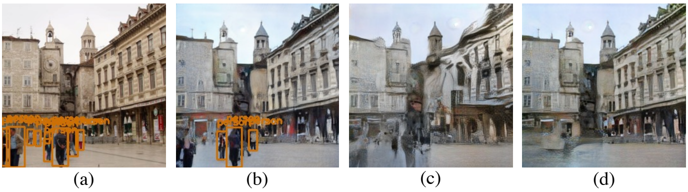


Figure 1: **Reverse Engineering Attack and Mitigations.** (a) Original image. Objects detected marked in orange (b) Reverse-engineered image using our attack. The reconstruction preserves semantic information. By (c) reducing the number of features or (d) selective suppression around private objects, we reduce the efficacy of the attack and improve privacy.

2 Related Work

The concept of reverse engineering local features has evolved over recent years as local descriptors play an increasingly important role. Prior work focused on better understanding the image features. Only recently have there been proposals towards leveraging this line of research to understand the privacy implications. Work towards discovering vulnerabilities and preventing attacks remains an emerging area of research.

Reconstructing Images from Sparse Local Features. Weinzaepfel et al. [19] demonstrated the feasibility of reconstructing images given SIFT [3] descriptors and keypoint locations, by finding and stitching the nearest neighbors in a database of patches. d'Angelo et al. [23] cast the reconstruction problem as regularized deconvolution to recover the image content from binary descriptors and keypoint locations (FREAK [20], ORB [18]). Kato and Harada [26] showed that it is possible to recover some of the structures of the original image from an aggregation of sparse local descriptors in bag-of-words (BoW) representation, even without keypoint locations. While the quality of reconstructed images from these methods is low, they allow clear interpretations of semantic content. Here, we demonstrate that reverse engineering attacks using CNNs reveal more image details and quantitatively analyze privacy implications for hand-crafted [3, 20], and machine-learned descriptors [51].

Reconstructing Images from Dense Feature Maps. Vondrick et al. [25] perform a visualization of HoG [10] features to understand its gaps for recognition tasks. To understand what information is captured in CNNs, Mahendran and Vedaldi [29] showed the inversions of CNN feature maps as well as a differentiable version of DenseSIFT [16] and HoG [10] descriptors using gradient descent. Dosovitskiy and Brox [34] directly model the inverse of feature extraction for HoG [10], LBP [4] and AlexNet [37] using CNNs, and qualitatively show better reconstruction results than the gradient descent approach [29]. They also show reconstructions from SIFT [3] features using descriptor, keypoint, scale, and orientation information. In our reconstruction, we use descriptors and keypoints only.

Modern Reverse Engineering Attacks. In the context of 3D point clouds and the AR/VR applications built on top of them, a common formulation of the reverse engineering attack is to synthesize scene views given the 3D reconstruction information. Recent work by Pittaluga et al. [49] showed that it is possible to reconstruct a scene from an arbitrary viewpoint from SfM models using the projected keypoints, sparse RGB values, depth, and descriptors. Our work extends this approach by considering only keypoints and descriptors.

Mitigations for Attacks on Sparse Local Features. For reverse engineering attacks on local features, one notable recent work [50, 54, 55] proposes using line-based features to obfuscate the precise location of keypoints in the scene to make the reconstruction difficult. The key idea is to lift every keypoint location to a line with a random direction, but passing through the original 2D [54] or 3D keypoints [50]. Since the feature location can be anywhere on a line, this alleviates privacy implications in the standard mapping and localization process. Shibuya et al. [55] later extended this approach for SLAM. Similarly, Dusmanu et al. [53] represent a keypoint location as an affine subspace passing through the original point, as well as augmenting the subspace with adversarial feature samples, which makes it more difficult for an adversary to recover original image content.

Mitigations on Raw Images. Other works try to alleviate the privacy concern by perturbing the images [45, 47, 28, 35, 38, 46, 48, 52]. One way of achieving this is to mask out or replace the parts of images (e.g., faces) that may contain private information [39, 45, 47]. Another stream of work focuses on encoding schemes or degrading images to prevent recognition of private image content [28, 35, 38, 46, 48, 52]. A few cryptographic methods using homomorphic encryption [14, 15, 40] have emerged, but they are computationally expensive and it is unclear how to apply them to complex applications such as localization.

Relationship to Adversarial Attacks on Neural Networks. Recent work has shown that it is possible to trick deep learning models with adversarial inputs to induce incorrect outputs [24, 30, 22, 41]. Conceptually, these adversarial attacks are similar to the mitigation strategies that we propose. But, unlike prior work, our insight is that inputs can be modified to induce incorrect outputs to *defend* against reverse engineering attacks.

3 System and Threat Definition

In this section, we define privacy, utility, the trade-offs, and define our privacy threat model.

Privacy. LINDDUN, a popular methodology in academic discussions, looks at the following privacy properties [17]: linkability, identifiability, non-repudiation, detectability, information disclosure, content unawareness, and policy. LINDDUN claims that whenever users share information, one or more of these privacy properties may be at risk. This leads to the notion that minimizing the amount of shared information improves privacy. However, precisely quantifying the impact on privacy is application-specific and can be implemented as a continuum, modulating the amount of information to be shared as required. In this work, references to privacy risk and/or threat applies specifically to reidentification risk, a direct result of the reverse engineering attack; we describe and evaluate the trade-offs in Section 5.2.

Utility. Utility captures the accuracy (or performance) of an application. Applications may have multiple utility functions for a well-rounded understanding of the operation. Utility often presents a trade-off with privacy as performance tends to increase with data size, e.g., ML training. We use feature matching recall as a proxy for localization accuracy (Section 5.2).

Privacy-Utility Trade-Off. Applying privacy-preserving techniques can adversely affect utility. Ideally, we want high utility and high privacy, but in practice there is a fundamental trade-off between the amount of information one is willing to share and the utility one receives from sharing it. In this work, the trade-off is between the localization accuracy (utility) and the images that may potentially be revealed (privacy). In certain cases where the definitions of utility and privacy are simple, this trade-off can be formalized and reasoned about analytically (e.g. k-anonymity [5]). In larger systems this is not possible and we must actively play roles of attacker and defender to model possible attacks and understand the potential risks to user privacy from reidentification. This is the role of a *privacy threat model* [2, 7, 9, 21, 56, 8, 17].

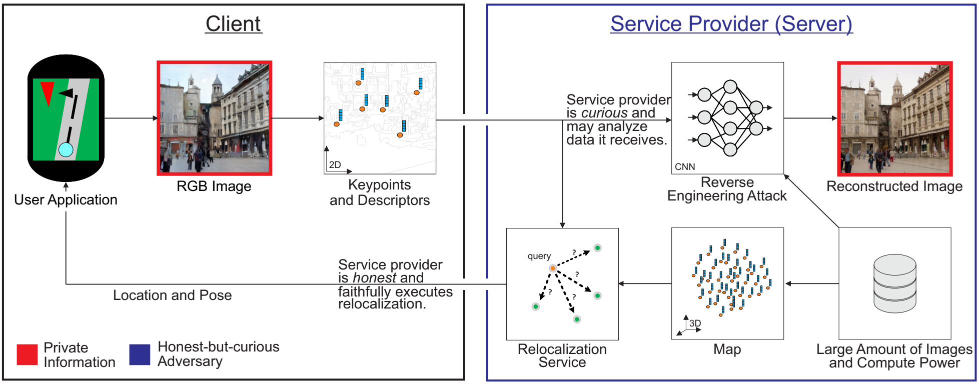

Client
User Application
RGB Image
2D
Keypoints and Descriptors
Location and Pose
Service provider is honest and faithfully executes relocalization.
🟥 Private Information
🟦 Honest-but-curious Adversary
Service Provider (Server)
Service provider is curious and may analyze data it receives.
CNN
Reverse Engineering Attack
Reconstructed Image
query
?
Relocalization Service
3D
Map
Large Amount of Images and Compute Power

Figure 2: **Privacy threat model for localization.** A client derives descriptors from RGB images and shares them with a service provider. The service provider is honest and faithfully executes localization by matching query descriptors against a map. But, the service provider may attempt to derive insight about the user. Our mitigation strategy is to minimize information shared between the client and the service provider to maximize privacy.

3.1 Privacy Threat Model

Building a privacy threat model is application specific. For localization, we use the LINDDUN "hard privacy" threat model [17]. LINDDUN proposes building a dataflow diagram of a system and marking data assets, adversaries, and potential attack vectors. These are used to audit against potential threats (described in LINDDUN) that impact privacy. We focus on identifiability, detectability, and information disclosure to audit potential reverse engineering attacks on RGB images. *Identifiability* checks if an adversary can identify items of interest. *Detectability* looks at whether an adversary can detect whether items exist or not. *Information disclosure* asks if private information is disclosed to an adversary without access. An adversary with an RGB image can observe information about each of these properties which poses a risk to privacy. Our goal is to prevent the adversary from having such access.

System Definition and Sensitive Data Assets. Figure 2 shows the components of our privacy threat model. Our system follows a client-server architecture to process localization requests. For localization, there are two primary data assets: (1) RGB images (2) feature descriptors. We prevent the sharing of RGB images which can leak private information. Descriptors are perceived as more private and more acceptable to share because they do not *directly* leak RGB information. The client derives feature descriptors (from RGB images) and shares them with the server to query its pose from a global map.

Adversary Definition and Potential Attacks. Our privacy threat model considers the service provider as an adversary (Figure 2) that is *honest-but-curious* [42]. This type of adversary is a legitimate participant in the system and executes the agreed upon service faithfully. But, while fulfilling the service, the adversary is *curious* and may use available data to learn information about the client. In our case, the adversary might reverse engineer the user's RGB images from feature descriptors. This is possible because the adversary has access to similar data (feature descriptors, source RGB images) and large scale compute resources. The adversary is capable of training deep-learning models (such as a reverse engineering model) to analyze user data in a reasonable amount of time. Our goal is to understand how to improve a client's protection against an honest-but-curious adversary capable of training

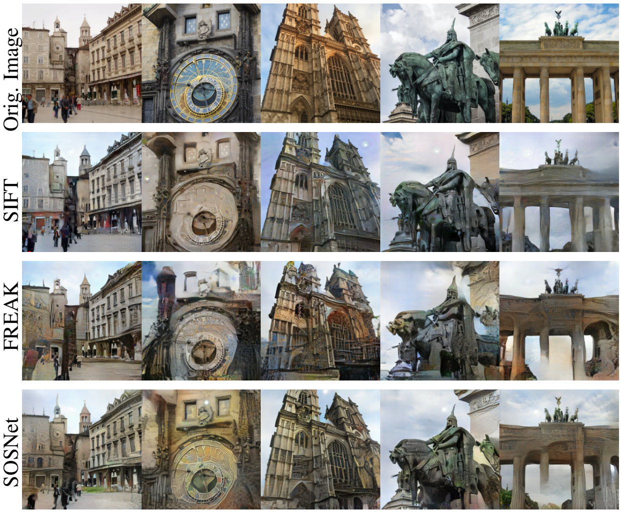


Figure 3: **Reverse Engineering Attack Results.** Top to bottom: ground truth and reconstructions from a max. of $1,000$ sparse SIFT, FREAK and SOSNet features. Reconstruction from only sparse local features reveals the original image information extremely well. Note: images show landmarks not included in the training data. Image attribution [11].

deep learning models to reverse engineer RGB images from feature descriptors.

4 Reverse Engineering Attack

This section defines the convolutional neural network models we use to craft our reverse engineering attack. As shown in Figure 2, this model takes sparse local features (keypoints and descriptors) as input and estimates the original RGB image.

Model Architecture. Given a user image $\mathbf{I}(i,j) \in \mathbb{R}^3$ and a derived sparse feature map $\mathbf{F}_{\mathbf{I},M}(i,j) \in \mathbb{R}^C$ containing C-dimensional local descriptors from the image $\mathbf{I}$ using a feature extractor M, we seek to reconstruct an image $\hat{\mathbf{I}}(i,j) \in \mathbb{R}^3$ from $\mathbf{F}_{\mathbf{I},M}$. The sparse feature map is assembled by starting with zero vectors and placing extracted descriptors at keypoint locations i,j. Our reverse engineering attack relies on a deep convolutional generator-discriminator architecture that is trained for each specific feature extraction method M. The generator G_M produces the reconstructed image: $\hat{I} = G_M(\mathbf{F}_{\mathbf{I},M})$ and follows a single 2-dimensional U-Net topology [32] with 5 encoding and 5 decoding layers as well as skip connections with convolutions. The discriminator D_M is a 6 layer convolutional network operating on top of G_M [31]. Please see the supplemental material for details. In order to adhere to our privacy threat model and in contrast to prior work by Pittaluga et al. [49], we do not use depth or RGB inputs and subsequently also do not make use of a VisibNet.

Loss Functions. We use 3 loss functions to train the reconstruction network. The *mean absolute error* (MAE) is the pixelwise L1 distance between the reconstructed and ground truth RGB images (Eq. 1). The *L2 perceptual loss* is measured as in Eq. 2 with ϕ_k being the outputs of a pre-trained and fixed VGG16 ImageNet model [13]. ϕ_k are taken after the ReLU layer k with $k \in \{2,9,16\}$. For the generator-discriminator combination, we use the *binary cross-entropy* (BCE) loss defined as in Eq. 3. Finally, we optimize the losses together as shown in Eq. 4 with α and β as scaling factors.

$$L_{mae} = \sum_{i,j} ||\hat{\mathbf{I}}(i,j) - \mathbf{I}(i,j)||_1 \tag{1}$$

$$L_{perc} = \sum_{i,j} \sum_{k=1}^{3} ||\phi_k(\hat{\mathbf{I}}(i,j)) - \phi_k(\mathbf{I}(i,j))||_2^2 \tag{2}$$

$$L_{bce} = \sum_{i,j} log(D_M(\mathbf{I}(i,j))) + log(1 - D_M(\hat{\mathbf{I}}(i,j))) \tag{3}$$

$$L_G = L_{mae} + \alpha L_{perc} + \beta L_{bce} \tag{4}$$

5 Evaluation

5.1 Experimental Setup

Sparse Local Features. For feature extraction from Section 4, we use SIFT [3], FREAK [20], and SOSNet [51] descriptors representing traditional and machine-learned variants. Keypoint locations for FREAK and SOSNet were detected using Harris corner detection [1]. For reconstruction, we use the SIFT detector for SIFT descriptors as in [49]; however, for image matching we use Harris corners as the SIFT detector performed poorly. We do not use additional information from Harris corner detector except keypoint locations in our experiments.

Training and Evaluation Data. We train our networks on $50,000$ images and their extracted sparse local features from MegaDepth's [43] training partition. For testing the attack, we sampled $9,800$ images from the MegaDepth testset that contain potentially private objects.

Network Training. A different reverse engineering model M is trained for 400 epochs for each descriptor type. The learning rate is initialized to 0.001 and 0.0001 for the generator and discriminator networks respectively. Learning rates are adjusted by the Adam optimizer [27].

5.2 Measuring Privacy and Utility

Measuring Privacy with SSIM. Our first metric for measuring privacy is structural similarity (SSIM), which measures the perceptual similarity between images. We use SSIM to evaluate how well the reverse engineering attack can recover visual information, i.e., to measure identifiability. SSIM looks at the **whole** image, which includes private and public information (e.g. people and buildings respectively). Measuring how well the whole image can be reconstructed includes the reconstruction quality of private regions.

Measuring Privacy by Object Detection. We use an object detector (YOLO v3 [44], with 80 classes) to measure semantic information from the reverse-engineered images. We compare object detection results on both the original and the reconstructed images. If an object's bounding box in the original image has at least 50% overlap with the reconstructed image of the same class label, we consider them a match. The more correspondence between objects in the original and the reconstructed image, the higher the risk to privacy.

Measuring Utility. To assess utility of local features when applying our mitigation strategies, we define an *image matching* task as a proxy for localization and investigate how the feature matching between two images deteriorates as we increase the privacy. Specifically, we generate corresponding image pairs from the 53 landmarks of the test split of the MegaDepth [43] dataset. For each landmark, we sample 50 pairs of images that have at least 20 covisible 3D points determined from a reference map built with COLMAP [36], resulting in $2,650$ image pairs. For each corresponding pair of images, we perform local correspondence matching using input features, and count the number of pairs with at least 20 inlier matches which we deem as successful. We refer to the proportion of image pairs that have been successfully matched as our matching recall, which we use as our utility measure (Table 2).

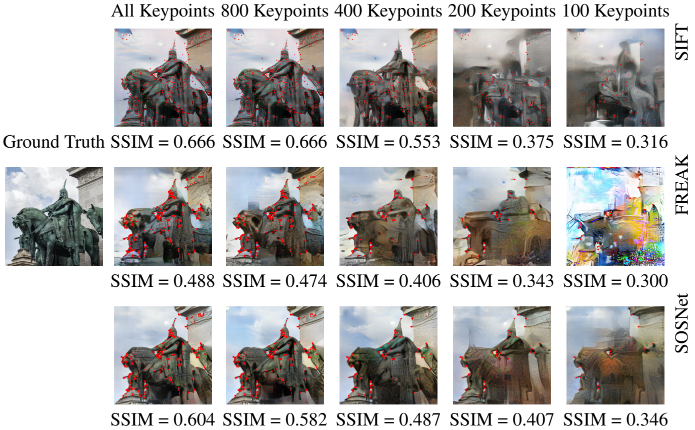


Figure 4: **Reverse engineering ablation study of reducing keypoints.** SIFT, FREAK and SOSNet reverse engineering results using $1,000$, 800, 400, 200, and 100 keypoints respectively, annotated in red. Reducing keypoints reduces the potency of the reverse engineering attack. Regions with higher densities of keypoints have better reconstruction quality.

Descriptor	SSIM	Detected Objects
SIFT [3]	0.675	32.58%
FREAK [20]	0.511	19.32%
SOSNet [51]	0.616	41.26%

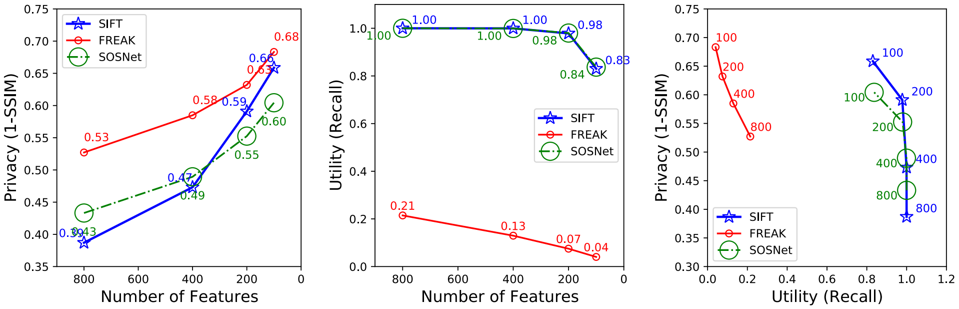
Table 1: **Privacy metrics of reverse-engineered images using** $1,000$ **keypoints.** Detected object percentage [44] is relative to number of objects detected in ground truth.

5.3 Reverse Engineering Attack

We first evaluate to what extent the reverse-engineering attack from Section 4 poses a reidentification risk to privacy. Examples of the reconstructions are shown in Figure 3 and the privacy metrics of the reverse-engineered images are given in Table 1. Reconstructions using FREAK [20] descriptors yield substantially poorer reconstruction quality and semantic content than SIFT [3] and SOSNet [51]. Despite differences in feature extraction techniques and descriptor sizes, all three descriptors are susceptible to the attack and yield reconstructions comparable to prior work [49] (please see supplemental material for detailed comparison to prior work), but notably without RGB or depth information as input. At a higher level, the results show that under controlled conditions the reverse engineering attack can introduce a reidentification risk of RGB image content. The results from Table 1 also show that the reverse-engineered images still allow an adversary to potentially detect and identify some objects that were present in the original images.

5.4 Mitigation by Reduction of Features

To improve privacy, our objective is to minimize the information shared by the client (Sec 3.1). We investigate how reducing the number of features increases privacy at the expense of utility. For each descriptor type, we retain a maximum of N top-scoring keypoints based on the detector response and vary N from 1000 to 100. For each value of N we then evaluate how well our reverse-engineering models perform. Qualitative results are given in Figure 4. We show the average privacy (measured by $1-$SSIM) of the reconstructed images vs. the num-



(a) (b) (c)

Figure 5: **Utility-Privacy Trade-Off when Varying the Number of Features.** Privacy increases when reducing the number of features where FREAK gives the best results. For utility, FREAK and SIFT gives the best results. SIFT gives the best overall trade-off.

Suppression	Privacy (Object Recall)		Utility (Matching Recall)	
	No	Yes	No	Yes
SIFT [3]	20%	**2.21%**	100%	**88%**
FREAK [20]	11%	1.29%	34%	28%
SOSNet [51]	28%	5.21%	100%	88%

Table 2: **Privacy-Utility Trade-Off for Suppression.** Object recall shows how many objects can be detected from the reverse engineered images compared to the original without and with suppression (lower is better). Matching recall shows how many images can be matched without and with selective feature suppression. SIFT gives the best overall trade-off.

ber of features in Figure 5a. The data shows the reconstruction SSIM degrades in as more keypoints are removed. For less than 300 features, SIFT gives better results than SOSNet. FREAK outperforms SIFT and SOSNet, and yields the best results in terms of privacy.

However, despite strong privacy results, FREAK has poor utility. In Figure 5b, we show how the utility changes. Here, FREAK gives the lowest utility, indicating that FREAK descriptors overall provide less useful information than SOSNet and SIFT. Interestingly, for SOSNet and SIFT the number of keypoints can be reduced to 200 by sacrificing only 2% performance. The trade-off between utility and privacy is shown in Figure 5c. Overall, we find that SIFT yields the best privacy-utility trade-off among the evaluated descriptor configurations on the Megadepth dataset. We note that these results do not preclude the possibility that other descriptor configurations (i.e., in terms of dimensionality, target dataset, and type) may achieve better results. Ultimately the ideal descriptor chosen will depend on the precise privacy and utility requirements necessitated by the localization service.

5.5 Selective Suppression of Features

Globally reducing image features reduces the potency of the reconstruction attack, but it reduces matching accuracy. We investigate how an object detector can help implement a more selective approach. We identify and mark sensitive regions in the images using bounding boxes produced by the YOLO v3 [44] object detector. We then suppress features in these regions. Finally, we apply our reverse-engineering attack and measure the detectable semantic information content in the images before and after reverse engineering (Table 2).

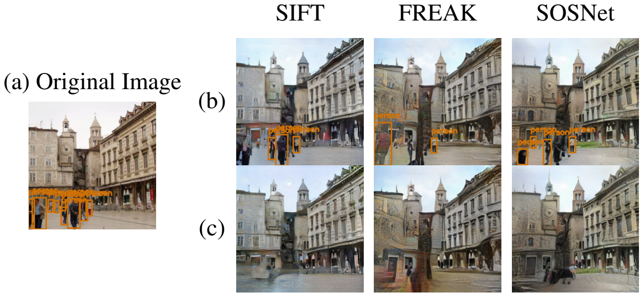


Figure 6: **Reverse Engineering after Selective Feature Suppression.** (a) Object detection on original image (b) Object detection on reverse-engineered images (max. 1000 keypoints) (c) Object detection on reverse-engineered images with feature suppression. All objects detected by the object detector without suppression are successfully removed with suppression.

Figure 6 shows a qualitative example of how selective feature suppression effectively defeats the object detector; the people detected in the original image do not appear nor are identifiable by the object detector in the reconstructed images. These results confirm our intuition that selective suppression can effectively preserve the privacy around a potentially sensitive region of interest (in our case semantic content of people in the image). Note that the quality of the overall image outside of the marked sensitive regions remains largely unaffected. Finally, the results show that features of private objects should not be shared in order to mitigate privacy risks posed by reverse engineering attacks.

Results for the privacy-utility trade-off of the suppression are given in Table 2. Under the evaluated experimental conditions, SIFT and SOSNet give better trade-offs than FREAK; these trends are consistent with the results from Section 5.4. Notably for SIFT the utility drops slightly, while the detected objects are almost eliminated.

6 Conclusion

This paper looks at the privacy of image-based localization systems. For the first time, we have shown a reverse engineering attack that operates in the real-world scenario, where only sparse local features are available to an honest-but-curious adversary. We found that our reverse engineering attack could reconstruct the original image with surprisingly good quality, posing a risk to privacy. We formulate a privacy threat model to review these threats and introduce two mitigation techniques and showed a trade-off between privacy and utility (measured by feature matching). We found that using an object detector to suppress objects slightly reduces matching accuracy (as a proxy for localization accuracy) but gives better privacy results (fewer reidentifiable objects). Finally, our analysis has shown that, among the descriptors we evaluate, the best overall privacy-utility trade-off can be achieved with SIFT, when compared to FREAK and SOSNet. Privacy (defined as reidentification risk through reverse engineering attacks as specifically described in this paper) may be preserved with the mitigation techniques described in this paper. Looking forward, our work provides initial experiments on some mitigation techniques the community may consider to further the privacy-aware descriptor-based applications research.

References

[1] Christopher G Harris, Mike Stephens, et al. "A combined corner and edge detector." In: *Alvey vision conference*. Vol. 15. 50. Citeseer. 1988, pp. 10–5244.

[2] Chris Salter et al. "Toward a secure system engineering methodolgy". In: *Proceedings of the 1998 workshop on New security paradigms*. 1998, pp. 2–10.

[3] David G. Lowe. "Object recognition from local scale-invariant features". In: *ICCV*. 1999.

[4] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns". In: *TPAMI* (2002).

[5] Latanya Sweeney. "k-anonymity: A model for protecting privacy". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (2002), pp. 557–570.

[6] Eldad Eilam. *Reversing: Secrets of Reverse Engineering*. USA: John Wiley & Sons, Inc., 2005. ISBN: 9780764574818.

[7] Suvda Myagmar, Adam J Lee, and William Yurcik. "Threat modeling as a basis for security requirements". In: *Symposium on requirements engineering for information security (SREIS)*. Vol. 2005. Citeseer. 2005, pp. 1–8.

[8] Paul Saitta, Brenda Larcom, and Michael Eddington. "Trike v. 1 methodology document [draft]". In: *URL: http://dymaxion. org/trike/Trike v1 Methodology Document-draft. pdf* (2005).

[9] Peter Torr. "Demystifying the threat modeling process". In: *IEEE Security & Privacy* 3.5 (2005), pp. 66–70.

[10] Qiang Zhu et al. "Fast human detection using a cascade of histograms of oriented gradients". In: *CVPR*. 2006.

[11] J. Miers. *Brandenburg Gate*. [Online; accessed February 1, 2021]. 2008. URL: https://commons.wikimedia.org/w/index.php?curid=22940398.

[12] Christian Cachin, Idit Keidar, and Alexander Shraer. "Trusting the cloud". In: *Acm Sigact News* 40.2 (2009), pp. 81–86.

[13] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *CVPR*. 2009.

[14] Zekeriya Erkin et al. "Privacy-preserving face recognition". In: *International symposium on privacy enhancing technologies symposium*. 2009.

[15] Ahmad-Reza Sadeghi, Thomas Schneider, and Immo Wehrenberg. "Efficient privacy-preserving face recognition". In: *International Conference on Information Security and Cryptology*. 2009.

[16] Ce Liu, Jenny Yuen, and Antonio Torralba. "Sift flow: Dense correspondence across scenes and its applications". In: *TPAMI* (2010).

[17] Mina Deng et al. "A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements". In: *Requirements Engineering* 16.1 (2011), pp. 3–32.

[18] Ethan Rublee et al. "ORB: An efficient alternative to SIFT or SURF". In: *ICCV*. 2011.

[19] Philippe Weinzaepfel, Hervé Jégou, and Patrick Pérez. “Reconstructing an image from its local descriptors”. In: *CVPR*. 2011.

[20] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. “Freak: Fast retina keypoint”. In: *CVPR*. 2012.

[21] Tony UcedaVelez. “Real world threat modeling using the pasta methodology”. In: *OWASP App Sec EU* (2012).

[22] Battista Biggio et al. “Evasion attacks against machine learning at test time”. In: *Joint European conference on machine learning and knowledge discovery in databases*. Springer. 2013, pp. 387–402.

[23] Emmanuel d’Angelo et al. “From bits to images: Inversion of local binary descriptors”. In: *TPAMI* 36.5 (2013), pp. 874–887.

[24] Christian Szegedy et al. “Intriguing properties of neural networks”. In: *arXiv preprint arXiv:1312.6199* (2013).

[25] Carl Vondrick et al. “Hoggles: Visualizing object detection features”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2013, pp. 1–8.

[26] Hiroharu Kato and Tatsuya Harada. “Image reconstruction from bag-of-visual-words”. In: *CVPR*. 2014.

[27] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[28] Daniel J Butler et al. “The privacy-utility tradeoff for remotely teleoperated robots”. In: *HRI*. 2015.

[29] Aravindh Mahendran and Andrea Vedaldi. “Understanding deep image representations by inverting them”. In: *CVPR*. 2015.

[30] Anh Nguyen, Jason Yosinski, and Jeff Clune. “Deep neural networks are easily fooled: High confidence predictions for unrecognizable images”. In: *CVPR*. 2015.

[31] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *arXiv preprint arXiv:1511.* (2015).

[32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *MICCAI*. Springer. 2015.

[33] Kim Wuyts and Wouter Joosen. “LINDDUN privacy threat modeling: a tutorial”. In: *CW Reports* (2015).

[34] Alexey Dosovitskiy and Thomas Brox. “Inverting visual representations with convolutional networks”. In: *CVPR*. 2016.

[35] Michael S Ryoo et al. “Privacy-preserving human activity recognition from extreme low resolution”. In: *arXiv preprint arXiv:1604.03196* (2016).

[36] Johannes Lutz Schönberger and Jan-Michael Frahm. “Structure-from-Motion Revisited”. In: *CVPR*. 2016.

[37] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Communications of the ACM* (2017).

[38] Nisarg Raval, Ashwin Machanavajjhala, and Landon P Cox. “Protecting visual secrets using adversarial nets”. In: (2017).

[39] Nishant Vishwamitra et al. “Blur vs. block: Investigating the effectiveness of privacy-enhancing obfuscation for images”. In: *CVPRW*. 2017.

[40] Ryo Yonetani et al. “Privacy-preserving visual learning using doubly permuted homomorphic encryption”. In: *ICCV*. 2017.

[41] N. Akhtar and A. Mian. “Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey”. In: *IEEE Access* 6 (2018), pp. 14410–14430. DOI: 10.1109/ACCESS.2018.2807385.

[42] Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha Chandrakasan. “{GAZELLE}: A low latency framework for secure neural network inference”. In: *USENIX*. 2018.

[43] Zhengqi Li and Noah Snavely. “Megadepth: Learning single-view depth prediction from internet photos”. In: *CVPR*. 2018.

[44] Joseph Redmon and Ali Farhadi. “Yolov3: An incremental improvement”. In: *arXiv preprint arXiv:1804.02767* (2018).

[45] Zhongzheng Ren, Yong Jae Lee, and Michael S Ryoo. “Learning to anonymize faces for privacy preserving action detection”. In: *CVPR*. 2018.

[46] Zhenyu Wu et al. “Towards privacy-preserving visual recognition via adversarial training: A pilot study”. In: *ECCV*. 2018.

[47] Tao Li and Lei Lin. “AnonymousNet: Natural Face De-Identification With Measurable Privacy”. In: *CVPRW*. 2019.

[48] Francesco Pittaluga, Sanjeev Koppal, and Ayan Chakrabarti. “Learning privacy preserving encodings through adversarial training”. In: *WACV*. 2019.

[49] Francesco Pittaluga et al. “Revealing scenes by inverting structure from motion reconstructions”. In: *CVPR*. 2019.

[50] Pablo Speciale et al. “Privacy preserving image-based localization”. In: *CVPR*. 2019.

[51] Yurun Tian et al. “Sosnet: Second order similarity regularization for local descriptor learning”. In: *CVPR*. 2019.

[52] Zihao W. Wang et al. “Privacy-Preserving Action Recognition Using Coded Aperture Videos”. In: *CVPRW*. 2019.

[53] Mihai Dusmanu et al. “Privacy-Preserving Visual Feature Descriptors through Adversarial Affine Subspace Embedding”. In: *arXiv preprint arXiv:2006.06634* (2020).

[54] Marcel Geppert et al. “Privacy Preserving Structure-from-Motion”. In: ECCV. 2020.

[55] Mikiya Shibuya, Shinya Sumikura, and Ken Sakurada. “Privacy preserving visual SLAM”. In: *arXiv preprint arXiv:2007.10361* (2020).

[56] MM Morana. *Wiley: Risk centric threat modeling: Process for attack simulation and threat analysis-tony ucedavelez, marco m. morana. Accessed on 09/05/2016*.