Adversarial Attack on Deep Product Quantization Network for Image Retrieval

Yan Feng,[1,2,*] **Bin Chen,**[1,2,*] **Tao Dai,**[1,2,†] **Shu-Tao Xia**[1,2]
[1]Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, China
[2]PCL Research Center of Networks and Communications, Peng Cheng Laboratory, Shenzhen, China
{y-feng18, cb17}@mails.tsinghua.edu.cn, daitao.edu@gmail.com, xiast@sz.tsinghua.edu.cn

Abstract

Deep product quantization network (DPQN) has recently received much attention in fast image retrieval tasks due to its efficiency of encoding high-dimensional visual features especially when dealing with large-scale datasets. Recent studies show that deep neural networks (DNNs) are vulnerable to input with small and maliciously designed perturbations (a.k.a., adversarial examples). This phenomenon raises the concern of security issues for DPQN in the testing/deploying stage as well. However, little effort has been devoted to investigating how adversarial examples affect DPQN. To this end, we propose product quantization adversarial generation (PQ-AG), a simple yet effective method to generate adversarial examples for product quantization based retrieval systems. PQ-AG aims to generate imperceptible adversarial perturbations for query images to form adversarial queries, whose nearest neighbors from a targeted product quantizaiton model are not semantically related to those from the original queries. Extensive experiments show that our PQ-AQ successfully creates adversarial examples to mislead targeted product quantization retrieval models. Besides, we found that our PQ-AG significantly degrades retrieval performance in both white-box and black-box settings.

Introduction

The massive application of large scale and high dimensional image data has attracted great attention to the study of efficient image retrieval methods. In order to strike the balance between storage and precision, approximate nearest neighbor (ANN) based retrieval methods (Datar et al. 2004; Gionis et al. 1999; Muja and Lowe 2009; G, T, and P 2006) have been widely studied. Among these methods, product quantization (PQ) based retrieval (Jégou, Douze, and Schmid 2013; Ge et al. 2013) becomes popular due to the efficiency and effectiveness of PQ for coding high-dimensional visual features. Recently proposed deep product quantization network (DPQN) based methods (Cao et al. 2016; Klein and Wolf 2019; Yu et al. 2018; Liu et al. 2019) combine the representative power of deep neural networks (DNNs) with PQ, and achieve impressive performance and efficiency.

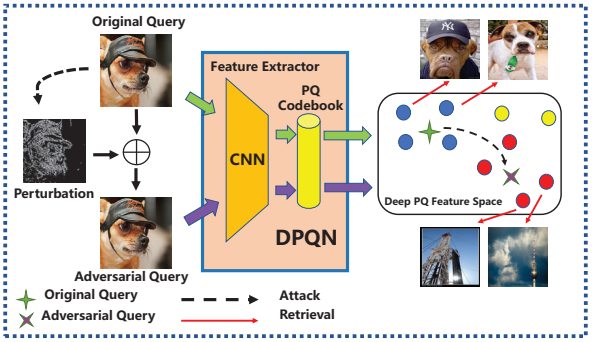


Figure 1: An example of adversarial attack against DPQN. Although the added perturbation is almost invisible, it successfully causes the query to shift a lot in deep PQ feature space, and fools the DPQN to retrieve semantically irrelevant images.

On the other hand, recent studies show that DNNs are vulnerable to input images with carefully designed yet imperceptible perturbations, i.e., adversarial examples, which can cause DNN classifier to produce confidently wrong predictions (Szegedy et al. 2014). To date, much attention has been drawn to crafting adversarial examples in image classification task, while little effort has been devoted to investigating how adversarial examples affect DPQN for image retrieval. This raises the concern of security issues of DPQN in practice. As shown in Fig. 1, DPQN based image retrieval system is successfully misled by an adversarial query image with imperceptible perturbations. Note that the well-known adversarial attacks for image classification can be interpreted as maximizing the classification error via tuning the input image according to the direction of gradients. However, attacking DPQN retrieval systems can be much more challenging than the classification task for two main reasons: 1) in an image retrieval task, a query image is searched through a database on visual features in the absence of class labels; 2) the optimization of adversarial attacks generally relies on back-propagation that requires differentiable oper-

[*]The first two authors contribute equally to this work.
[†]Corresponding author: Tao Dai

ations, while DPQN consists of product quantization operation, which is indifferentiable.

Motivated by the above observations, we propose product quantization adversarial generation (PQ-AG), a simple yet effective method to craft adversarial examples for deep product quantization models. Given a clean query image and a targeted product quantization retrieval model, our proposed PQ-AG method can generate adversarial queries with small and well-crafted perturbations to fool the targeted model to retrieve semantically irrelevant images from the database. Specifically, we first formulate the objective as a feature separation problem, which pushes the features of adversarial query away from those of the original neighbors. Furthermore, to deal with the indifferentiable issue of PQ, we alternatively propose to maximize the difference of the codewords assignment probability distribution between the adversarial query and the original one, where the distributions can be estimated based on the similarity between features and codewords.

Our main contributions can be summarized as follows:

- We propose a simple yet effective product quantization adversarial generation (PQ-AG) method to mislead the DPQN based image retrieval systems. To the best of our knowledge, this is the first attempt of adversarial attack design targeting on the DPQN based image retrieval system.
- In order to tackle the indifferentiable problem of product quantization, we propose a novel strategy to design an adversarial query by perturbing the probability distribution of codewords assignments for a clean query.
- We evaluate PQ-AG on a variety of datasets and model structures. Experimental results demonstrate the effectiveness and transferability of our method in both white-box and black-box settings.

Related Work

We give the formal definition of PQ and briefly review recent advances of PQ based retrieval methods applied in lager scale and high dimension datasets. Then we also provide a brief introduction to some related works on adversarial attack.

PQ based Image retrieval systems

To efficiently conduct the approximate nearest neighbor (ANN) search for image retrieval in large scale databases, existing mainstream approaches can be divided into two two categories: 1) hashing based methods 2) PQ based methods. Hashing method aims at learning a hash function, which maps the original data points to a binary code. The main advantage of hashing methods is the fast-retrieval speed due to the computation by hamming distance and low storage overhead. Unlike hashing methods that can only produce a few distinct distances, PQ based methods can better describe the distance between data points, whose goal is to employ K-means individually in each subspace of original data points or their transformed ones. Next, we will briefly describe the core of PQ based image retrieval methods.

Formally, suppose that $\mathcal{D} \subseteq \mathbb{R}^D$ is a database containing N data points, a given vector $\mathbf{x} \in \mathcal{D}$ is first partitioned into M sub-vectors with equal dimension D/M, i.e. $\mathbf{x} = (\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_M})$. Then each subvector $\mathbf{x_m}$, $m \in [M] \triangleq \{1, 2, \ldots, M\}$, is quantized into the K centroids within a sub-codebook $\mathcal{C}_m = \{\mathbf{c_{m1}}, \ldots, \mathbf{c_{mK}}\}$ in each subspace uniquely as

$$q(\mathbf{x_m}) = \arg\min_k ||\mathbf{x_m} - \mathbf{c_{mk}}||_2^2, \tag{1}$$

where $\mathbf{c_{mk}}$ is the k-th centroid in the m-th sub-codebook. Therefore, every data point $\mathbf{x}$ can be approximately represented by its corresponding centroids, i.e.,

$$\mathbf{x} \approx q(\mathbf{x}) = (q(\mathbf{x_1}), ..., q(\mathbf{x_M})).$$

Note that PQ can generate a Cartesian product codebook $\mathcal{C} = \mathcal{C}_1 \times \mathcal{C}_2 \times \ldots \mathcal{C}_M$ with an exponential size. The cost of the storage is low with $\mathcal{O}(DK)$ and the retrieval speed is fast due to the use of look-up tables.

In the retrieval stage, PQ-based methods conduct approximated nearest neighbor (ANN) based on similarity measure between two images in feature space. There are two main types of similarity measures, i.e., symmetric distance computation (SDC) and asymmetric distance computation (ADC), to approximate Euclidean distance computation between the query $\mathbf{y}$ and the vector $\mathbf{x}$ in the database, which is defined as follows:

$$d_{\text{SDC}}(\mathbf{y}, \mathbf{x}) \approx d(q(\mathbf{y}), q(\mathbf{x})) = \sqrt{\sum_{m=1}^{M} d(q(\mathbf{y_m}), q(\mathbf{x_m}))^2},$$

$$d_{\text{ADC}}(\mathbf{y}, \mathbf{x}) \approx d(\mathbf{y}, q(\mathbf{x})) = \sqrt{\sum_{m=1}^{M} d(\mathbf{y_m}, q(\mathbf{x_m}))^2},$$

where $d(\cdot, \cdot)$ denotes the Euclidean distance.

Recently, many PQ based methods (Jégou, Douze, and Schmid 2013; Ge et al. 2013; Wang et al. 2014; Kalantidis and Avrithis 2014; Babenko and Lempitsky 2014; Wang et al. 2016; Yu et al. 2017; Wu et al. 2017; Li et al. 2017; Wang and Zhang 2019) have been developed for fast image retrieval. To obtain better retrieval performance, deep product quantization network (DPQN) based methods (Cao et al. 2016; Klein and Wolf 2019; Liu et al. 2019; Yu et al. 2018) have been recently proposed by combining PQ and deep convolutional neural network. Cao et al. (2016) gave a step forward in this direction and proposed the deep quantization network (DQN) to capture good hashing based image representations under a similarity-preserving and a product quantization loss. Later, deep product quantization (DPQ) (Klein and Wolf 2019) was developed by cluster optimization of PQ with the supervised signal and the parameters update of the fully-connected layer simultaneously. Liu et al. (2019) proposed the deep triplet quantization (DTQ) to learn deep quantization models by introducing a novel similarity triplet loss in the training process. To better incorporate PQ in a neural network, product quantization network (PQN) (Yu et al. 2018) adds a soft product quantization layer to the neural network and obtains a clever codewords assignment according to the similarity between the features, which

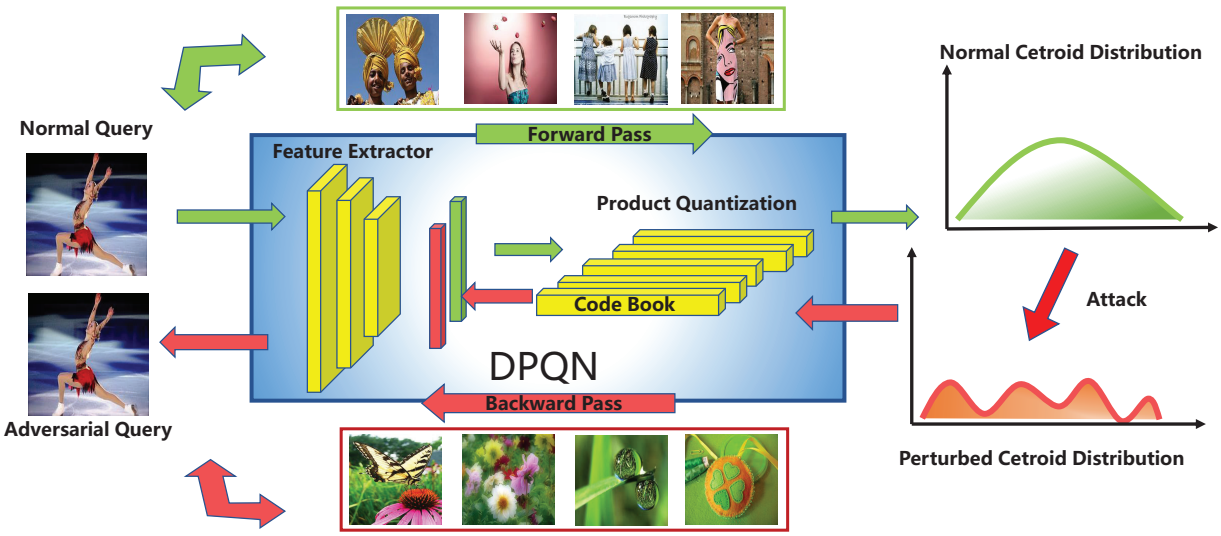


Figure 2: The overall pipeline of the proposed PQ-AG method against DPQN based retrieval systems. A query is first fed into the CNN model, then the normal centroid distribution is computed based on the cosine similarity between the deep feature and centroids of the codebook. The goal of PQ-AG is to perturb the distribution, thus disrupting the subsequent retrieval operation. Gradients of the perturbation will be back-propagated to update the adversarial query.

mitigates the over-fitting problem and consistently outperforms other methods. Despite of many DPQN based retrieval methods, little effort has been devoted to the security issue of DPQN for image retrieval, which is the main reason why we investigate how adversarial examples affect DPQN.

Adversarial Examples

Recent studies show that deep neural networks (DNNs) are vulnerable to input with small and maliciously designed perturbations, i.e., adversarial examples. Depending on the access permission to the information of the neural network, including the neural structure and parameters, adversarial attacks can be roughly divided into two types: white-box (Szegedy et al. 2014; Goodfellow, Shlens, and Szegedy 2015) and black-box attacks (Carlini and Wagner 2017; Goodfellow, Shlens, and Szegedy 2015; Bhagoji et al. 2018). White-box attacks can obtain better performance, since they can easily access to the gradient information of DNNs. Among them, Szegedy et al. (2014) recently showed that some targeted neural networks can be easily fooled by crafted adversarial examples. Other attack algorithms like fast gradient sign method (FGSM) based methods (Goodfellow, Shlens, and Szegedy 2015; Madry et al. 2018) also obtain remarkable performance. Instead, black-box attacks (Carlini and Wagner 2017; Goodfellow, Shlens, and Szegedy 2015; Moosavi-Dezfooli et al. 2017; Bhagoji et al. 2018) are more challenging, since they cannot access to the information of DNNs. Besides these attacks on image classification task, some other methods focus on the adversarial attack on other tasks, including image captioning (Chen et al. 2017) and semantic segmentation (Xie et al. 2017).

However, little effort has been devoted to the image retrieval task. To the best of our knowledge, two related works, i.e., hash adversary generation (HAG) (Yang et al. 2018) and UAA-GAN (Zhao et al. 2019), both focus on adversarial attack on image retrieval. Nevertheless, there exist two main differences compared with our method. 1) Our work is a first attempt to consider the adversary generation for product quantization based image retrieval, while HAG and UAA-GAN focus on hashing based adversarial perturbation generation and generative network design for conventional retrieval systems, respectively; 2) our work focuses on designing a simple yet effective attack method in product quantization feature space, which is more challenging, while HAG and UAA-GAN concentrate on attacking in deep feature space.

Product Quantization Adversarial Generation

In this section, we would demonstrate how product quantization adversarial generation (PQ-AG) algorithm generates adversarial queries to fool the DPQN based retrieval method into retrieving semantically irrelevant images. As shown in Fig. 2, our PQ-AG can generate adversarial query by perturbing the centroid distribution in product quantization space to attack DPQN.

Formulation of the Overall Objectives

In an image retrieval task, adversarial query generation algorithms for a targeted retrieval system usually consider the following problem: for a given query $\mathbf{y}$, the goal of adversarial query generation is to generate its corresponding adversarial example $\hat{\mathbf{y}}$, whose retrieval results are semantically irrelevant to $\mathbf{y}$. Denote $F(\cdot,\cdot)$ as a feature-dependent transformation functions whose output contains the semantic information of the original feature. We further denote $d(\cdot,\cdot)$ as a metric to measure the similarity between $F(\mathbf{y})$ and $F(\hat{\mathbf{y}})$. Therefore, the core idea of generating adversarial queries is

to expand the distance as defined by metric d between the semantic features $F(\hat{\mathbf{y}})$ and $F(\mathbf{y})$ for adversarial and clean queries.

To this end, we can formulate the adversarial query generation problem as follows:

$$\begin{aligned} \min_{\hat{\mathbf{y}}} \ & \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = -d\left(F(\mathbf{y}), F(\hat{\mathbf{y}})\right) \\ \text{s.t.} \ & ||\mathbf{y} - \hat{\mathbf{y}}|| \leq \eta, \end{aligned} \tag{2}$$

where $d(\cdot, \cdot)$ is a similarity measure function, e.g. Euclidean distance, $||\cdot||$ can be any norm specified by the user, e.g., ℓ_0, ℓ_1, or ℓ_∞-norm, and $\eta > 0$ denotes the magnitude of the attack.

For example, if we set $F(\cdot, \cdot)$ to be a CNN feature extractor $F_{\text{CNN}}(\boldsymbol{.})$ and $d(\cdot, \cdot)$ to be Euclidean distance, then the objective function (2) can be formulated as:

$$\begin{aligned} \min_{\hat{\mathbf{y}}} \ & \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = -||F_{\text{CNN}}(\mathbf{y}), F_{\text{CNN}}(\hat{\mathbf{y}})||_2^2 \\ \text{s.t.} \ & ||\mathbf{y} - \hat{\mathbf{y}}|| \leq \eta. \end{aligned} \tag{3}$$

In practise, (3) represents the attack on a general deep feature based retrieval system. Therefore, we adopt (3) as the baseline method in our experiments.

In order to design a more successful attack against DPQN, we need have a closer look at its retrieval process. As shown in Fig. 2, DPQN mainly consists of a feature extractor and a PQ codebook. During retrieval process, the input query is first fed into the feature extractor, which is usually a pre-trained convolutional neural network (CNN). Afterwards, the features are quantized as the closest centroids from the codebook, and the quantized feature can be written as:

$$F(\mathbf{y}) = q\left(F_{\text{CNN}}(\mathbf{y})\right), \tag{4}$$

where $q(\cdot)$ denotes the product quantization operation as (1), which will be also referred to as hard centroid assignment operation in the subsequent parts. The retrieval results can then be determined as the nearest neighbors of the product-quantized feature from the database.

It is clear that product quantization plays a key role in the above DPQN based retrieval process. Therefore, attacking on the quantized feature could be an effective way for adversarial query generation.

The specific form of (2) against DPQN can thus be formulated as:

$$\begin{aligned} \min_{\hat{\mathbf{y}}} \ & \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = -||q\left(F_{\text{CNN}}(\mathbf{y})\right) - q\left(F_{\text{CNN}}(\hat{\mathbf{y}})\right)||_2^2 \\ \text{s.t.} \ & ||\mathbf{y} - \hat{\mathbf{y}}|| \leq \eta. \end{aligned} \tag{5}$$

The next step is to optimize the objective in (5). Since the function $q(\cdot)$ in (5) involves the indifferentiable operation, i.e., hard codewords assignment, it is infeasible to directly optimize (5) based on back propagation. To solve this problem, we alternatively propose the soft codewords assignment, whose details are shown in the following sections.

PQ based Attack via Centroid Distribution Perturbation

The proposed PG-AG seeks to generate adversarial examples in PQ feature space. From the above analysis, the main challenge of our PQ-AG lies in the indifferentiable issue of PQ. To solve this issue, we alternatively estimate the probability distribution of codewords assignment of a clean query based on the hard version (1), i.e., one-hot encoding.

Attack on the Peak of the Centroid Distribution (Type I Attack) Denotes $\mathbf{z}$ as the l_2-normalized feature extracted from a pre-trained CNN for $\mathbf{y}$. Then the sub-vectors of $\mathbf{z}$ is assigned to the optimal centroids of sub-codebook $\{\mathbf{c_{mk}}\}_{\mathbf{m=1,k=1}}^{\mathbf{M,K}}$, and the indices $(b_1, b_2, \ldots, b_m)$ of $\mathbf{z} = (\mathbf{z_1}, \mathbf{z_2}, ..., \mathbf{z_M})$ can be specifically computed by

$$b_m = \arg\max_k \ \langle \mathbf{z_m}, \mathbf{c_{mk}} \rangle, \ m \in [M]. \tag{6}$$

In order to solve the gradient-vanishing problem, we adopt the soft centroid assignment first proposed in (Yu et al. 2018). In detail, we calculate the soft probability distribution $\hat{\mathbf{p}}_\mathbf{m} = (\hat{p}_{m1}, \hat{p}_{m2}, \ldots, \hat{p}_{mK})$ of assigning the subvector $\hat{\mathbf{z}}_\mathbf{m}$ of $\hat{\mathbf{z}} = F_{\text{CNN}}(\hat{\mathbf{y}})$ to the centroids $\{\mathbf{c_{mk}}\}_{\mathbf{k=1}}^{\mathbf{K}}$ based on the respective cosine similarities as follows.

$$\hat{p}_{mk} = \frac{e^{\langle \hat{\mathbf{z}}_\mathbf{m}, \mathbf{c_{mk}} \rangle}}{\sum_{k'} e^{\langle \hat{\mathbf{z}}_\mathbf{m}, \mathbf{c_{mk'}} \rangle}}, \ k \in [K]. \tag{7}$$

It is easy to obtain the ground-truth probability distribution $\mathbf{p_m}$, i.e., one-hot encoding of the clean query $\mathbf{y}$ according to (6). If we take the cross-entropy loss as our metric d to measure the similarity between $\hat{\mathbf{p}}_\mathbf{m}$ and $\mathbf{p_m}$, then the objective function (2) can be specifically formulated as:

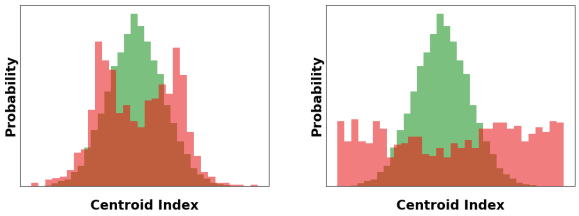
$$\begin{aligned} \min_{\hat{\mathbf{y}}} \ & \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = \sum_m -d(\hat{\mathbf{p}}_\mathbf{m}, \mathbf{p_m}) \\ & = \sum_m \sum_k \log(\hat{p}_{mk}) \mathbb{1}(k = b_m) \\ & = \sum_m \sum_k \log\left(\frac{e^{\langle \hat{\mathbf{z}}_\mathbf{m}, \mathbf{c_{mk}} \rangle}}{\sum_{k'} e^{\langle \hat{\mathbf{z}}_\mathbf{m}, \mathbf{c_{mk'}} \rangle}}\right) \mathbb{1}(k = b_m), \\ \text{s.t.} \ & ||\mathbf{y} - \hat{\mathbf{y}}|| \leq \eta, \end{aligned} \tag{8}$$

where $\mathbb{1}(\cdot)$ is an indicator function.

Note that the above objective (8) bears a resemblence to the loss commonly used in untargeted attack for classification tasks, where the $\hat{p}_{mk}$ corresponds to the probability of classifying the adversarial example to the k_{th} class and b_m corresponds to the true label. As shown in Fig. 3a, this kind of attack effectively decrease the probability of selecting the peak centroid of original distribution.

Attack on the Overall Centroid Distribution (Type II Attack) Type I attack, however, may still have some limitations due to the following two key observations for symmetric and asymmetric retrieval attacks.

- In the phase of the symmetric retrieval, both the feature of query and database images will be quantized. Fig. 3 shows that attacking on the peak will merely push the original centroid distribution towards a multi-peak distribution, of which the peaks correspond to the originally high-probability centroids. Moreover, those centroids with high probabilities, i.e., centroids with small



(a) Type I Attack (b) Type II Attack

Figure 3: Centroid distribution under Type I and Type II attack. Normal distrbution is plotted with green color while the perturbed distribution with red color. Compared with Type I attack, Type II attack disrupts the overall distribution instead of simply focusing on the peak.

angular difference, are very similar. Therefore, for a symmetric retrieval system, optimizing (8) may simply replace the hard-quantized centroids of a clean query $\mathbf{y}$ with slightly different ones, thus unable to effectively increasing the quantization error or attack effect.

- In the phase of the asymmetric retrieval, only the feature of database images will be quantized. Therefore the similarity score or relevance between a query feature $\mathbf{z} = (\mathbf{z_1}, \mathbf{z_2}, ..., \mathbf{z_M})$ and a database feature $\mathbf{x} = (\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_M})$ is defined as

$$s(\mathbf{z}, \mathbf{x}) = \sum_{m=1}^{M} \langle \mathbf{z_m}, \mathbf{c_{mb_m}} \rangle,$$

where $b_m = \arg\max_k \langle \mathbf{x_m}, \mathbf{c_{mk}} \rangle,\ m \in [M]$. Note that the scores are clearly determined by the inner product pairs $\{\langle \mathbf{z_m}, \mathbf{c_{mk}} \rangle \mid m \in [M], k \in [K]\}$ and implicitly determined by the distribution $\{p_{mk}\}_{m=1,k=1}^{M,K}$ computed by (7). Consequently, it's crucial to distort the overall distribution instead of focusing purely on attacking the peak of the distribution of a clean query $\mathbf{y}$ as in (8).

The above two observations motive us to pay more attention to perturbing the overall soft-quantized distributions $\{p_{mk}\}_{m=1,k=1}^{M,K}$ of a clean query $\mathbf{y}$. To this end, we replace the one-hot distribution in (8) with the soft distribution $\{p_{mk}\}_{m=1,k=1}^{M,K}$ as (7) to learn an adversarial query $\hat{\mathbf{y}}$. Formally, we reformulate the objective (2) as follows:

$$\begin{aligned} &\min_{\hat{\mathbf{y}}} \ \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = \sum_m \sum_k p_{mk} \log(\hat{p}_{mk}), \\ &\text{s.t. } ||\mathbf{y} - \hat{\mathbf{y}}|| \leq \eta. \end{aligned} \tag{9}$$

Note that optimizing (9) is equivalent to maximize the KL divergence between the real distributions $\{p_{mk}\}_{m=1,k=1}^{M,K}$ and adversarial distributions $\{\hat{p}_{mk}\}_{m=1,k=1}^{M,K}$. In this way, we can completely push the feature away from those centroids with high probabilities. Especially for the asymmetric attack, we have expanded the distance between the adversarial distribution $\{\hat{p}_{mk}\}_{m=1,k=1}^{M,K}$ and the original one, thus differentiating the similarity scores for $\hat{\mathbf{y}}$ and $\mathbf{y}$. As illustrated in Fig. 3b, Type II Attack leads to a more disordered distribution, which is harder to defend. More results about our Type I and II Attacks are shown in the experiment section.

Experiments

In this section, we evaluate our proposed product quantization adversarial generation (PQ-AG) algorithm on two public benchmark datasets: **CIFAR-10** and **NUS-WIDE**. We first introduce the datasets, evaluation metric and implementation details. Then we present and analyze the experimental results in detail.

Datasets and Metrics

CIFAR-10 (Krizhevsky and Hinton 2009) contains 60000 color images of size 32×32 divided into 10 classes, each of which contains 6000 images. Following conventional configuration (Yu et al. 2018), (Cao et al. 2016), the training of targeted product quantization retrieval model is conducted on the training set containing 50000 images, while the test set is randomly divided into 9000 images as a database and 1000 images as queries.

NUS-WIDE (Chua et al. 2009) is a large scale dataset for multilabel classification tasks. NUS-WIDE consists of 269648 images and each image is assigned with one or multiple labels related to 81 concepts. We select the subset of 186577 images associated with the 10 most popular concepts. The training set and test set are separated as default setting in (Chua et al. 2009). For the generation of adversarial queries, 1000 images are randomly selected from the test set as query images, while the remaining images are used as database images.

Evaluation Metrics We adopt the standard metric in image retrieval tasks, mean Average Precision (mAP), to evaluate the effectiveness of our methods. For the fairness of comparison, we evaluate the retrieval performance in both SDC and ADC settings, where SDC retrieval systems symmetrically quantize both query and database images and ADC systems quantize query images only.

Implementations

Unless otherwise stated, the targeted deep PQ models are constructed with the most common configuration. First, the base models (e.g. AlexNet, VGG) are pretrained on the evaluated datasets. Afterwards, we extract features from the top layer of the pretrained networks for each of the database images. Then the feature vectors are partitioned into M sub-vectors and K-means clustering is conducted for each of the M subspace, obtaining the respective codebooks $\{\mathbf{c_{mk}}\}_{\mathbf{m=1,k=1}}^{\mathbf{M,K}}$.

All the experiments are conducted based on the PyTorch framework.

PQ-AG methods are implemented based on AdverTorch (Ding, Wang, and Jin 2019) Framework. For adversarial query generation, we adopt PGD (Kurakin, Goodfellow, and Bengio 2017) to optimize our proposed algorithms. The learning rate is set to 0.01, and each query is trained for 5 iterations. The adversarial purturbations are restricted within 8 by L_{inf} (pixel values are clipped in [0, 255]).

		CIFAR-10						NUS-WIDE					
		Alex			VGG			Alex			VGG		
		16bits	24bits	36bits	16bits	24bits	36bits	16bits	24bits	36bits	16bits	24bits	36bits
SDC	Clean	81.0	83.9	84.1	83.6	86.7	86.9	65.7	66.0	66.1	72.1	73.6	73.6
	Basic	27.5	22.1	26.7	19.9	14.8	13.0	47.7	47.2	46.8	37.0	34.7	33.9
	APD	13.7	15.7	17.5	16	10.9	9.1	31.5	29.8	28.9	30.9	29.2	28.1
	AOD	**12.3**	**14.8**	**17.2**	**13.6**	**9.5**	**8.0**	**28.9**	**28.9**	**28.8**	**28.5**	**26.7**	**25.7**
ADC	Clean	81.3	84.5	84.6	84.0	86.3	86.7	66.2	66.1	65.9	73.2	73.4	73.3
	Basic	25.3	22.1	26.7	18	13.4	12.2	46.5	46.3	46.2	35.0	33.5	33.0
	APD	10.3	14.4	17.2	13.5	9.4	8.2	30.3	28.9	**28.3**	28.6	27.5	27.0
	AOD	**10.2**	**13.2**	**16.6**	**12.0**	**8.5**	**7.4**	**28.2**	**28.1**	28.5	**26.3**	**25.0**	**24.6**

Table 1: The white-box attack results for a different number of bits of two models on CIFAR-10 and NUS-WIDE. Lower mAP means better attack performance.

Attack Evaluation

We evaluate the effectiveness of our proposed PQ-AG algorithm containing two types of attacks, i.e., Type I: **A**ttak on the **P**eak of Centroid **D**istribution (APD) and Type II: **A**ttak on the **O**verall Centroid **D**istribution (AOD) on the datasets discussed above. The basic attack (denoted as basic) introduced in (3) is also evaluated in the experiments. In total, the experiments are conducted on three well-known network structures (AlexNet, VGG, and ResNet, each with multiple models). The performance of the proposed methods under white-box and-black box settings will be examined in the following experiments, respectively.

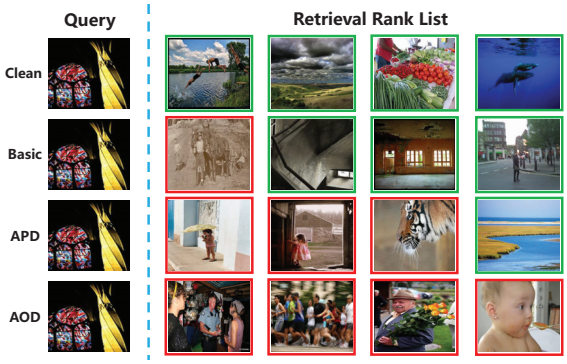


Figure 4: Examples of retrieval results for the respective attack, where irrelevant retrieval results are labeled with red borders.

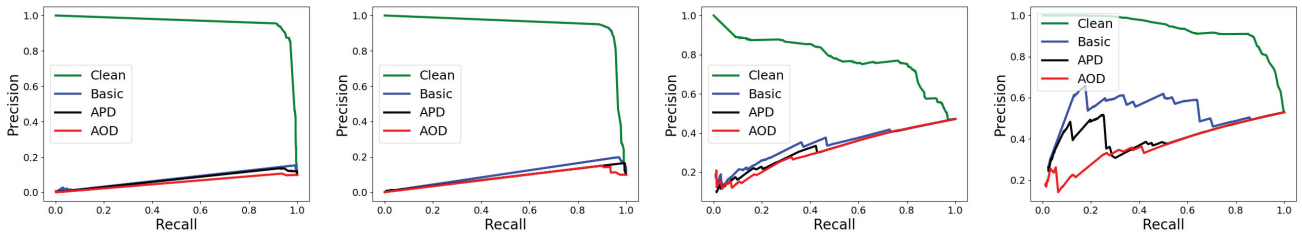
White-Box Attack We first evaluate the white-box effectiveness of the proposed attacks against basic attack across encoding bits. To make a more comprehensive comparison, we use AlexNet and VGG16 in white-box experiments. The mAP for the datasets under various attacks with the encoding bits ranging from 16 bits to 32 bits, are shown in Tab. 1. Clearly, it can be seen from Tab. 1 that modern product quantization based image retrieval models are extremely vulnerable to adversarial queries. Even under the simplest basic attack, the mAP value drops significantly by 57.7% on average. Then APD attack consistently outperforms the Basic Attack across models and encoding bits for both CIFAR-10 and NUS-WIDE datasets, which reveals that for product quantization systems, manipulating the original quantization results is an effective means of increasing the quantization error, thus result in the precision dropping drastically. AOD attack shows its priority over APD attack, and continue to diminish the mAP for most settings. Such an improved result means that, despite its success in image classification attacks, decreasing the probability for the most probable centroids (class in the case of image classification) may not be the best solution to image retrieval attacks. Instead, we should disrupt the overall distribution of the codebook. Some visualization can be found in Fig. 4.

Precision Recall (PR) curves for the proposed methods on CIFAR-10 and NUS-WIDE are shown in Fig. 5. For the fairness of comparison and better visualization, all the curves are calculated with 16-bits quantization. We can see from Fig. 5 that, different from the original one, PR curves for adversarial examples have a monotone increasing tendency and become furthest away from the original curves for the top retrieval regions, where the recall precision is low. This demonstrates that the proposed attacks completely disrupt the semantic information in the feature space, and effectively pushes the features of adversarial queries away from the originally relevant images in the database.

We also demonstrate the effectiveness of the proposed attacks on the state-of-the-art DPQN. More specifically, we adopt the method proposed in (Yu et al. 2018), and jointly train the retrieval model with PQ code-book in an end-to-end manner. The attack results against state-of-the-art retrieval models on CIFAR-10 are shown in Tab. 2. As can be seen from Tab. 2, despite the improved performance on clean data, the jointly trained models still suffer from the lack of robustness. In fact, comparing Tab. 2 with Tab. 1, we could find that the jointly trained models are even less robust than the pretrained models against our PQ-AG attacks. These results indicate that robustness should also be included in evaluation when designing DPQNs in the future.

Black-Box Attack We try to evaluate the Type II PQ-AG attack in the black-box settings from two aspects as follows:

- **Bits Transferibility:** For a given targeted model with var-



(a) AlexNet on CIFAR-10 (b) VGG16 on CIFAR-10 (c) AlexNet on NUS-WIDE (d) VGG16 on NUS-WIDE

Figure 5: PR curves for AlexNet and VGG16 on CIFAR-10 and NUS-WIDE, respectively.

		16bits	24bits	36bits
Alex	Clean	87.9 / 87.8	89.6 / 89.4	89.7 / 89.5
	Basic	17.8 / 12.8	12.1 / 9.5	9.7 / 8.4
	APD	15.0 / 9.5	9.7 / 6.6	7.4 / 5.7
	AOD	11.5 / 8.8	8.2 / 6.5	6.5 / 5.8

Table 2: White-box attack results for the jointly trained model on CIFAR-10.

ious code lengths, the generated adversarial queries can be transferable.

- **Model Transferibility:** For various model architectures, the generated adversarial queries can be transferable.

		16bits	24bits	36bits
Alex	Clean	81.0 / 81.3	83.9 / 84.5	84.1 / 84.6
	16bits	12.3 / 10.2	13.6 / 13.4	17.0 / 15.4
	24bits	15.5 / 14.3	14.8 / 13.2	16.7 / 15.7
	36bits	17.6 / 19.2	15.9 / 15.1	17.2 / 16.6
VGG	Clean	83.6 / 84.0	86.7 / 86.3	86.9 / 86.7
	16bits	13.6 / 12.0	9.6 / 8.6	8.4 / 7.8
	24bits	13.8 / 13.3	9.5 / 8.5	8.3 / 7.7
	36bits	13.5 / 11.8	9.4 / 8.4	8.0 / 7.4

Table 3: Evaluation results for bits transferability.

The bits transferability of PQ-AG can be found in Tab. 3. Each row header of Tab. 3 represents the code length of the model for generating the clean and adversarial queries, and each column header represents the code length of the targeted model. The experiments are conducted on CIFAR-10 for both AlexNet and VGG16 models. As can be seen from the table, the mAP values (SDC/ADC) in each column are really close, which verifies the transferability of PQ-AG across bits. This could be attributed to the fact that the centroids in the codebooks across code lengths are essentially cluster centers from the same feature space. Therefore, centroids in a larger code book could be seen as interpolation among the ones in a smaller codebook. As a result, perturbing the centroid distribution for one of the code lengths could lead to the corresponding distributions of all code lengths to be disrupted.

		Res18	Res34	Res50	VGG11	VGG13	VGG16
SDC	Clean	87.7	89.2	87.1	84.3	85.0	83.6
	Res18	9.5	36.8	46.0	66.9	67.5	64.5
	Res34	30.8	10.8	33.8	62.5	62.2	60.0
	Res50	38.6	32.7	10.5	65.4	65.4	62.4
	VGG11	71.0	73.0	73.5	12.6	61.6	64.1
	VGG13	61.1	64.0	64.0	50.1	12.2	49.7
	VGG16	60.2	58.6	61.7	53.4	49.8	13.6
ADC	Clean	87.8	89.5	87.8	84.7	85.2	84.0
	Res18	7.7	38.5	47.2	67.9	69.0	66.0
	Res34	30.3	8.3	34.2	63.6	63.6	61.3
	Res50	39.9	34.5	8.3	66.5	67.0	64.3
	VGG11	72.1	74.2	75.3	9.8	63.3	65.8
	VGG13	63.0	65.7	66.0	52.0	9.6	51.6
	VGG16	61.3	60.8	63.3	55.2	51.6	12.0

Table 4: Evaluation results for model transferability.

Tab. 4 demonstrates the transferability of PQ-AG among different CNN models. We conduct the experiments on CIFAR-10 for ResNet and VGG models. As before, each row header represents the model, form which the adversarial queries are generated, while each column header represents the structure of the targeted model. For both SDC and ADC product quantization retrieval systems, we divide the table into four quarters. The diagonal quarters are designed to evaluate the transferability of PQ-AG among the models from the same families (e.g. VGG11, VGG13, VGG16 are all from the VGG family), ensuring the models share a certain amount of similarity. Therefore, we can see from Tab. 4 that the mAP drops by a large margin of 68.5% and 51.6% respectively for ResNet and VGG models in this two quarters on average. However, adversarial queries in other two quarters are generated against one kind of models and designed to fool models from completely different families. Therefore, the task is obviously much more challenging. Even so, PQ-AG still manages to decrease the mAP by 24.9% on average in all these cases.

Conclusion

A novel PQ-AG algorithm is developed for generating adversarial queries to fool the deep product quantization retreieval systems. With the estimation of the centroid distribution, we propose a differentiable loss for back propagation to generate adversarial queries. Extensive experiments are conducted on two public datasets based on multiple network

structures, demonstrating that PQ-AG can effectively generate adversarial queries that induce the PQ based retrieval models into retrieving semantically irrelevant results. In the meantime, the transferability of the generated queries is also evaluated under a variety of settings, revealing that PQ-AG also works in black-box setting.

Acknowledgement

This work is supported in part by the National Key Research and Development Program of China under Grant 2018YFB1800204, the National Natural Science Foundation of China under Grant 61771273, the China Postdoctoral Science Foundation under Grant 2019M660645, the R&D Program of Shenzhen under Grant JCYJ20180508152204044, and the research fund of PCL Future Regional Network Facilities for Large-scale Experiments and Applications (PCL2018KP001).

References

Babenko, A., and Lempitsky, V. S. 2014. Additive quantization for extreme vector compression. In *CVPR*.

Bhagoji, A. N.; He, W.; Li, B.; and Song, D. 2018. Practical black-box attacks on deep neural networks using efficient query mechanisms. In *ECCV*.

Cao, Y.; Long, M.; Wang, J.; Zhu, H.; and Wen, Q. 2016. Deep quantization network for efficient image retrieval. In *AAAI*.

Carlini, N., and Wagner, D. A. 2017. Towards evaluating the robustness of neural networks. In *SP*.

Chen, H.; Zhang, H.; Chen, P.; Yi, J.; and Hsieh, C. 2017. Show-and-fool: Crafting adversarial examples for neural image captioning. *arXiv: 1712.02051*.

Chua, T.; Tang, J.; Hong, R.; Li, H.; Luo, Z.; and Zheng, Y. 2009. NUS-WIDE: a real-world web image database from national university of singapore. In *CIVR*.

Datar, M.; Immorlica, N.; Indyk, P.; and Mirrokni, V. S. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*. ACM.

Ding, G. W.; Wang, L.; and Jin, X. 2019. AdverTorch v0.1: An adversarial robustness toolbox based on pytorch. *arXiv preprint arXiv:1902.07623*.

G, S.; T, D.; and P, I. 2006. Nearest-neighbor methods in learning and vision: Theory and practice. *ch.3, MIT Press*.

Ge, T.; He, K.; Ke, Q.; and Sun, J. 2013. Optimized product quantization for approximate nearest neighbor search. In *CVPR*.

Gionis, A.; Indyk, P.; Motwani, R.; et al. 1999. Similarity search in high dimensions via hashing. In *Vldb*.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. In *ICLR*.

Jégou, H.; Douze, M.; and Schmid, C. 2013. Product quantization for nearest neighbor search. *TPAMI*.

Kalantidis, Y., and Avrithis, Y. 2014. Locally optimized product quantization for approximate nearest neighbor search. In *CVPR*.

Klein, B., and Wolf, L. 2019. End-to-end supervised product quantization for image search and retrieval. In *CVPR*.

Krizhevsky, A., and Hinton, G. 2009. Learning multiple layers of features from tiny images.

Kurakin, A.; Goodfellow, I. J.; and Bengio, S. 2017. Adversarial examples in the physical world. In *ICLR*.

Li, L.; Hu, Q.; Han, Y.; and Li, X. 2017. Distribution sensitive product quantization. *TCSVT*.

Liu, B.; Cao, Y.; Long, M.; Wang, J.; and Wang, J. 2019. Deep triplet quantization. *arXiv preprint arXiv:1902.00153*.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards deep learning models resistant to adversarial attacks. In *ICLR*.

Moosavi-Dezfooli, S.; Fawzi, A.; Fawzi, O.; and Frossard, P. 2017. Universal adversarial perturbations. In *CVPR*.

Muja, M., and Lowe, D. G. 2009. Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1)*.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I. J.; and Fergus, R. 2014. Intriguing properties of neural networks. In *ICLR*.

Wang, J., and Zhang, T. 2019. Composite quantization. *TPAMI*.

Wang, J.; Wang, J.; Song, J.; Xu, X.-S.; Shen, H. T.; and Li, S. 2014. Optimized cartesian k-means. *TKDE*.

Wang, X.; Zhang, T.; Qi, G.; Tang, J.; and Wang, J. 2016. Supervised quantization for similarity search. In *CVPR*.

Wu, X.; Guo, R.; Suresh, A. T.; Kumar, S.; Holtmann-Rice, D. N.; Simcha, D.; and Yu, F. 2017. Multiscale quantization for fast similarity search. In *NeurIPS*.

Xie, C.; Wang, J.; Zhang, Z.; Zhou, Y.; Xie, L.; and Yuille, A. L. 2017. Adversarial examples for semantic segmentation and object detection. In *ICCV*.

Yang, E.; Liu, T.; Deng, C.; and Tao, D. 2018. Adversarial examples for hamming space search. *IEEE transactions on cybernetics*.

Yu, L.; Huang, Z.; Shen, F.; Song, J.; Shen, H. T.; and Zhou, X. 2017. Bilinear optimized product quantization for scalable visual content analysis. *TIP*.

Yu, T.; Yuan, J.; Fang, C.; and Jin, H. 2018. Product quantization network for fast image retrieval. In *ECCV*.

Zhao, G.; Zhang, M.; Liu, J.; and Wen, J.-R. 2019. Unsupervised adversarial attacks on deep feature-based retrieval with GAN. *arXiv: 1907.05793*.