Convolutional Neural Network-based image retrieval with degraded samples

Dang Thanh Vu
Dept of Electronics and Computer Engineering
Chonnam National University
GwangJu, South Korea
dtvu1707@gmail.com

***Abstract*—Over the past decade, convolutional neural networks (CNNs) have been applied extensively on various tasks related to image. Given an input image, a CNN model will investigate the content and deduce the representation of this image using a model's structure built from hidden neurons. This representation analyzes data semantically, which helps to solve semantic issues, such as image retrieval. To verify the above viewpoint, this study addresses the problem of using features learned from a CNN model to perform image retrieval. To more emphasize the efficiency of learned features, I consider degraded images and their enhanced version as queries and search for similar ones in the gallery set. Data augmentation is also applied to increase the number of images in the gallery. The experiments are conducted on a multi-view dataset, smallNorb. Experimental results are reported both in quantity and quality.**

Keywords—Image retrieval, deep learning, data augmentation, image processing.

I. Introduction

Image analysis based on deep neural networks has been becoming a discipline in recent years. Especially, CNN [1] models have been deployed in plenty of fields of computer vision and image processing. Compared to traditional image processing methods, CNN-based approaches usually achieve better results and generalization. Instead of manual search over hyperparameters space in traditional methods, those of CNNs are encapsulated in a training process, which is an optimization procedure. Although training deep learning models is computationally expensive so far, recent advances in computing technologies (e.g., GPUs, computing cloud) have partially alleviated the difficulty in training neural networks. Moreover, model testing and application deployment are more favorable than ever with the help of many deep learning frameworks.

It is common sense that the distributed representation [2] of a multilayer model helps it to capture variation and retain the meaningful features in the context defined by the loss function. CNNs have further attained locally translational invariance by the share-weight mechanism. This mechanism also introduced feature detectors that extract similar features across the spatial dimension. Content-base image retrieval makes use of these feature detectors to seek the samples in the gallery, which has the same context either in vision level or semantic level.

The quality of the query has considerably affected the retrieval results. A high-quality query would make the retrieval less difficult, and the result in the gallery will be a similar one to the query with high confidence. However, it is rare in practice that the query is exactly analogous to one of the samples existing in the gallery. Moreover, the visual quality may be different between query images and samples in the gallery because of cross-domain (e.g., camera, encoding, environment). Also, there is a case that one would like to retrieve downgraded images to gain better knowledge based on similar results in the gallery. This study aims to address the above challenges by considering features extracted from a CNN model that already trained on a high-quality dataset, while the degraded image and its enhancement will be used in the testing phase.

II. Related Works

A. Image Retrieval

Before coming content-based methods, traditional image search engines are highly dependent on index multimedia visual metadata, such as titles and tags [3]. However, this indirect method might not be inconsistent with the visual content since the visual information is usually discarded. In content-based visual retrieval, there are two deficiencies; they are visual gap and semantic gap. The first one refers to the visual quality of query the image that does not match to that of samples in the gallery set. The semantic gap indicates the failure in describing the high-level semantic concept of a retrieval model.

There has been extensive research devoting to the two above challenges. Some of the very first solutions are using hand-craft features such as Bag of Word or SIFT, SURF descriptor [4]. However, such features require careful foreground extraction and image enhancement, which usually has no general way on a large-scale dataset.

Deep learning has become a core component in the top-performing methods for many image retrieval approaches. Albert et al. proposed an end-to-end learning model of deep visual representations for image retrieval using R-MAC descriptor and Siamese network [5]. Noh et al. introduced an attentive local feature descriptor that employed attention mechanisms for large-scale landmark image retrieval [6]. Bag of Word features and fine-tune CNN were employed in the study of [7] to address hard examples and enhance the performance of image retrieval.

B. Data augmentation and image processing

Data augmentation is a simple technique to increase the number of data by transforming available data. Based on the techniques, the dataset will become diverse and appropriate to the specific tasks. It is usually reported that using data augmentation helps to increase the performance and strengthen the generalization of learning models. However, an inappropriate data augmentation method will generate out-of-distribution samples that are directly harmful to the learning model. For that reason, one might carefully design which techniques are suitable for their specific problem or use auto data augmentation [8]. For example, one can use geometric transform and image scaling on classification problems. Additionally, noisy and downgraded samples may help with adversarial attacking problems.

In this article, I use image processing techniques such as geometric transformation, histogram matching [9] to do data augmentation. Meanwhile, I use noise models such as (e.g., Gaussian, periodic noise) and low pass filter to generate degraded samples and test the ability of the learned CNN model to these noisy data. The enhanced data will also experiment.

III. PROPOSED FRAMEWORK

A. *Feature extraction*

This section describes the proposed framework comprised of features extracted from the trained model and the metric used to measure the similarity between a query and image in the gallery.

First of all, I trained a classification model on the gallery set and utilize it to extract feature vectors for retrieval tasks. In this study, I considered ResNet18 [], a well know CNN architecture, as the classification model. ResNet has been successfully employed or a backbone model in plenty of research in the field of computer vision, such as image classification, object detection, and image segmentation. For a brief explanation, ResNet architecture consists of two core parts, feature extraction and class probability calculation. This network organizes information flowing from one layer to the next with the use of shortcuts so that the gradient does not vanish when traveling through a deep process. Besides, the classification mechanism in ResNet is vanilla, in which a series of fully – connected (FC) layers is used to model a non – linear classifier with softmax class probability

$$\hat{p}_k = P\left(\hat{Y} = k \middle| \boldsymbol{X} = x\right) = \frac{\exp_k\left(f_\theta(x)\right)}{\sum_{k'=1}^{K} \exp_{k'}\left(f_\theta(x)\right)} \qquad (1)$$

Where f_θ is the classification model parameterized by θ, taking an image x as the input and producing the score (logit) vector, and k is the index of class.

Neural networks ensemble distributed representation, where middle and early layers extract lower features then combine them to cultivate global features at the last layer [2]. Neural networks trained on a dataset for classification purposes can be used in various other tasks. For example, ResNet trained on the ImageNet dataset is widely used as the pre-trained model for another dataset. It is also deployed as a backbone network of U-Net like for segmentation or object detection. In general terms, meta-learning [], or shift domain is a valid characteristic of neural networks that enhances their learning ability and applications. Inspiring from the idea above, this study analyzes that feature vectors output from a neural network via a classification task could also be used for image retrieval. Particularly, I extracted the values output from the last convolutional block of ResNet and considered those as feature vectors. A feature vector represents an image that registers to a gallery and also used as the key of a query image. The pipeline is explained in Figure 1.

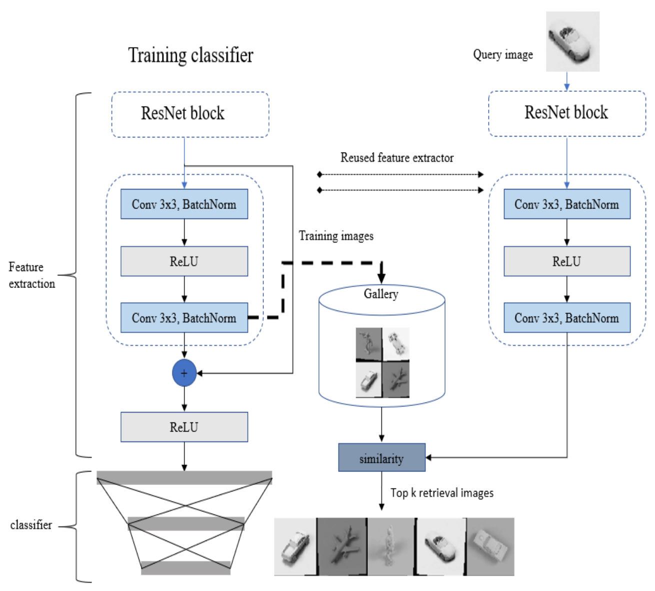


Figure 1. Overall framework. The pipeline includes two steps: training a classifier and image retrieval. The backbone model is ResNet, which is added 3 FC layers for classification. After training the network, training images will be used to build the gallery that is the collection of feature vectors extracted from the feature extraction part. Given a query image, the feature extractor will be reused to drive its corresponding feature vectors, and similarity will be then calculated between each sample in the gallery and the query, and finally return top k retrieval results.

B. *Similarity*

Once a gallery has been constructed from the collection of extracted feature vectors of training data, retrieval images will be queried by calculating the similarity between its feature vector and all samples in this gallery. In this work, I consider two similarities; they are Euclidean distance and cosine similarity. Given a vector $\boldsymbol{x}$ driven from the feature extractor of a query image and a candidate vector $\boldsymbol{y}$ in the gallery, the Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{y}$ is given in formula 2

$$d(\boldsymbol{x}, \boldsymbol{y}) = \|x - y\|_2 \qquad (2)$$

Seeking for top k smallest $\boldsymbol{y}$ using the Euclidean distance is analogous to finding k nearest neighbors. Another popular metric is cosine similarity, as given in formula 3.

$$d(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x}^{\boldsymbol{T}}\boldsymbol{y}}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|} \qquad (3)$$

Cosine similarity is generally used as a metric for measuring distance when the magnitude of the vectors does not matter. Two vectors are considered to close to each other if its Cosine similarity goes to 1.

Basically, representation space formulated by neural network layers is driven to minimize an objective function. However, it is not theoretically proved that this space complies with human visual. In another word, two similar images in the human perspective are not always in one cluster in the representation space. Moreover, adversarial samples, such as noisy images, easily make a model behave peculiarly [10]. Therefore, this study aims to show which cases a corrupt image violates the consistent in the representation space acquired by Euclidean distance and cosine similarity.

C. Degrading model

This section describes the degrading models, which are used in this study.

- Injecting noise

Noise is the most common factor that affects the performance of a learning model. In this study, I used three noise types; they are Gaussian noise, salt-pepper noise and speckle noise.

Gaussian noise is a typical noise that is usually assumed when the noise model is unknown:

$$g(x,y) = f(x,y) + \epsilon(x,y), \quad (4)$$

where $g(x,y)$ is the corrupted image generated from an image $f(x,y)$ and Gaussian noise $\epsilon \sim N(\mu,\sigma)$, in this study, I set $\mu = 0$, and $\sigma = 0.1$.

Salt and Pepper noise is also known as impulse noise caused by sharp and sudden disturbances:

$$g(x,y) = (1-m) \times f(x,y) + m \times \epsilon(x,y), \quad (5)$$

where $m \sim B(p = 0.9)$ indicates that only 10% of the image is injected with white noise $\epsilon = 1$ or black noise $\epsilon = 0$, where $\epsilon \sim B(p = 0.5)$.

Another noise model that considered in this work is speckle noise:

$$g(x,y) = f(x,y) + f(x,y) \times \epsilon(x,y), \quad (6)$$

where Uniform noise $\epsilon \sim U(a,b)$ is multiplied with the original image, in this study, I set $a = -0.1$, and $b = 0.1$.

Lowpass filters are usually applied to remove noise, yet each filter only well responds to a specific kind of noise. Although noise removal has also been an extensive field of research with various success, this study does not aim to noise removal methods. Therefore an adequate denoising autoencoder will be deployed as a unified model to eliminate three noise types mentioned above.

- Resolution reduction

It is a common case when test data has resolution relatively lower than training images. For example, users capture images with a low-resolution camera, or the main object is not focused as well as image is compressed for online transmission. To simulate this case, I apply multiresolution analysis by using wavelet decomposition, where an image's resolution is decreased 2 or 4 times. Formally,

$$\begin{aligned} f(x,y) &= \frac{1}{\sqrt{MN}}\sum_m\sum_n W_\phi(j_0,m,n)\phi_{(j_0,m.n)}(x,y) \\ &+ \frac{1}{\sqrt{MN}}\sum_{i=H,V,D}\sum_{j=j_0}^{\infty}\sum_m\sum_n W_\psi^i(j,m,n)\psi_{j,m,n}^i(x,y), \end{aligned} \quad (7)$$

where $W_\phi(j_0,m,n)$ is a smooth or lower resolution component, $\phi_{(j_0,m.n)}(x,y)$, $\psi_{j,m,n}^i(x,y)$ respectively is scaling and wavelet function [9]. Even though I fully model multiresolution analysis in the experiments below, the diagonal, vertical and horizontal components could not be attained in practice, so completely reconstructing high-resolution images is unaffordable. To this extend, in this study, a super-resolution network is considered to interpolate higher resolution images from the lower ones.

- Motion blurring

Motion blurring is a typical factor that frequently encounters when using handy cameras. Formally, motion blurring can be attained using a degraded model in the frequency domain [9]

$$g(x,y) = FFT^{-1}\{F(u,v) \times H(u,v)\} \quad (8)$$

$$H(u,v) = \frac{T}{\pi(ua+vb)}\sin\big(\pi(ua+vb)\big)e^{-j\pi(ua+vb)}, \quad (9)$$

where T, a, b are hyperparameters that control directions and magnitude of blurring. Similar to the case of noise models, motion blurring can be reverted with the degraded model yet not practical. Instead of inverted filtering, I train a network to deblur corrupted images.

IV. EXPERIMENTS

A. Dataset

smallNorb dataset is established for experiments in 3D object recognition from shape [10]. This database contains 96k grayscale images of 50 figures belonging to 5 categories. The objects were captured by two cameras under 6 lighting conditions, 9 elevations (30 to 70 degrees every 5 degrees), and 18 azimuths (0 to 340 every 20 degrees). In this study, the training set is composed of 5 instances of each category (from 4 to 9), and the test set contains the remaining 5 instances (from 0 to 5). The size of an image is 96×96, and intensity was normalized from -1 to 1. I used data augmentation on the training set to increase the observed cases, the techniques applied in this study were contrast and lighting adjustment, geometry transform (randomly translate 5 pixels and rotate 5 degrees) as well as horizontal flip, as illustrated on Figure 2.

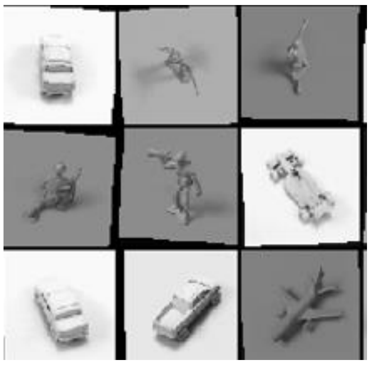


Figure 2. Data augmentation on smallNorb dataset

B. Experimental settings

First of all, I trained a classification model on the augmented training dataset, the network used here was ResNet34 where the last FC layer was replaced by a new FC layer only having 5 units corresponding to 5 classes. Since ResNet was developed to be capable of working with a color image, its first layer requires an input of 3 channels, so as to apply ResNet on a grayscale image, I manipulated a 1×1 convolutional layer to convert from 1 channel input to 3 channels input before feeding it through ResNet. The network was trained for 200 epochs with batch size of 64, the initial learning rate was set to 0.001 but decreased half every 40 epochs, and the cross-entropy loss function was minimized by Adam optimizer with default settings.

In terms of the autoencoder model, the network included an encoder part where 5 consecutive convolutional blocks were used to compose a corrupted image; each block had a 3×3 convolutional layer with padding of 1 and stride step of 2, followed by ReLu activation function and batch normalization. Likewise, the decoder part involved 5 continuos deconvolution blocks; each block had a 3×3 transposed convolutional layer [11] with padding of 1 and stride 2, output padding was set to 1 to guarantee that the spatial size twice times increased per block. Instead of ReLu, I used leaky ReLu as activation function on the decoder part for numerical stability, and batch normalization was also applied. However, the last layer employed tanh activation to ensure the output value range from -1 to 1. The architecture is shown in Figure 3. I used Adam optimizer to minimize the MSE reconstruction loss between ground truth image and output from autoencoder, and other configurations were set as in the case of training the classification model.

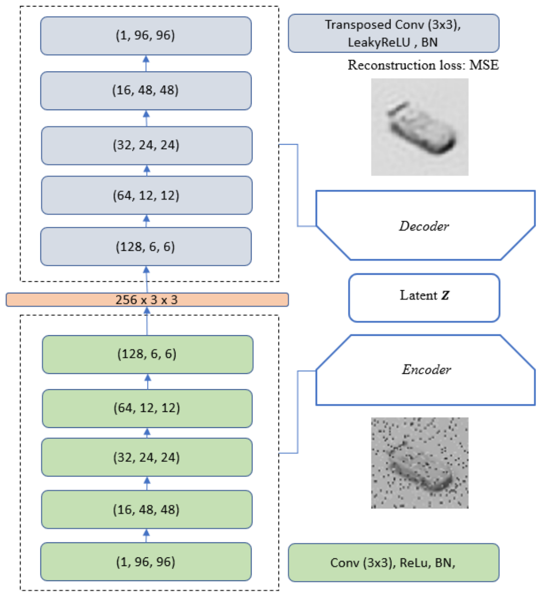


Figure 3. Autoencoder's architecture: the tuple inside each layer indicates the input channel and spatial size.

Regarding super-resolution CNN (SRCNN), I trained a network that interpolates a higher resolution image from the lower one. A low-resolution image was first applied nearest neighbor to upscale to the desired size, then fed through SRCNN, a network that kept the spatial dimension unchanged across layers. In particular, I constructed a network involving 3 convolutional layers, and each convolutional layer had padding size dependent on the designed kernel size. Note that SRCNN required only a single image as input [13], so the reconstruction loss is MSE between interpolated image and ground truth.

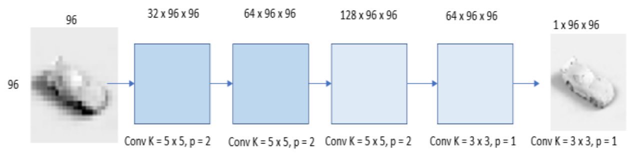


Figure 4. Super-resolution network: the first three convolutional layers had a kernel size of 5×5, and padding 2, the rest had a kernel size of 3×3 and padding 1.

I conducted all experiments on a Linux server with the processor of Intel Xeon v5 and Nvidia Titan GPU of 12 GB RAM. The main programing language was Python, whose auxiliary libraries were Pytorch and OpenCV. The source code is available at https://github.com/Ka0Ri/Image_retrieval_Net.git

C. *Experimental results*

The model achieved the classification accuracy of 88.73%, where the most misclassification belongs to class car and truck, as shown on the confusion matrix in Figure 5. One the trained model was ready, I extracted feature vectors and performed retrieval; details have been explained in section III. A. The gallery was built from the training data without augmentation, while a set of query images is the validation set. First, the retrieval results were reported in the ideal case, where test images were assumed that captured by the same settings with training images. Next, I considered the degraded images that were added noise, motion blurred and reduced resolution, as query images for the retrieval process. Figure 6 shows those corrupted images. Finally, the enhanced images attained from autoencoder and SRCNN were used to test the retrieval system.

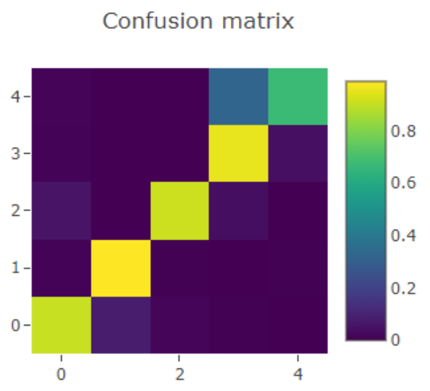


Figure 5. Confusion matrix. 1: animal, 2: human, 3: plane, 4: truck, 5: car.

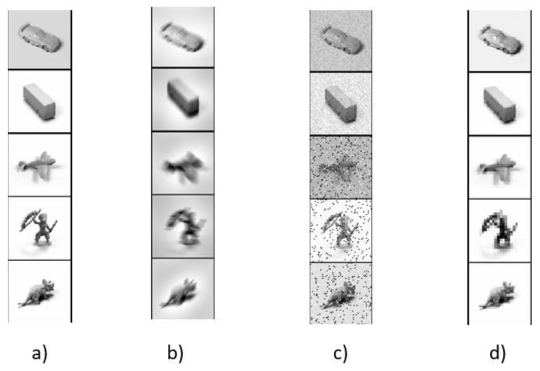


Figure 6. Degraded images: a) original, b) motion blurring, c) noisy, d) resolution reduction.

I used top k matching accuracy to evaluate the proposed framework. It measures the ratio that the class of given image matches with the class of one of k retrieval returns. Formally, given an image x, and $S(x)$ is a set containing k retrieval images predicted from the model, the top k matching accuracy is calculated by

$$m = \frac{\sum_i^n \delta(x_i)}{n}, \quad (8)$$

Where $\delta(x_i) = 1$ if exist a $y \in S(x_i)$ such that $label(x_i) = label(y)$, while $\delta(x_i) = 0$ if such y does not exist. Table 1 summarizes the retrieval accuracy of all considered cases.

Table 1. Retrieval performance

	Top 1	Top 5	Top 10
Ideal query			
L2 metric	89.42	91.82	92.32
cosine sim	89.47	91.82	92.30
Noisy query/enhanced			
L2 metric	39.95/55.38	42.41/60.90	42.84/64.41
cosine sim	40.24/55.37	42.10/60.62	42.91/61.86
Motion blurry/enhanced			
L2 metric	72.23/74.80	77.76/80.32	79.36/81.33
cosine sim	72.44/74.81	77.84/79.95	78.67/81.55
Low resolution/enhanced			
L2 metric	79.47/81.01	84.14/85.37	84.94/86.35
cosine sim	79.36/81.08	83.81/85.06	84.58/86.06

The experimental results in Table 1 show that the retrieval framework still worked well in cases of motion blurring and low resolution but severely affected by noisy samples. We hypothetically commented that representation vectors learned from CNN is not change much when morphing (shape) objects, but global factors such as noise cause tremendous distinction. This suggestion followed the principle of CNN's layers, where the global feature vectors driven from the last layer could not be maintained with noisy input. The enhancing models took effect in cases of noisy samples, so the retrieval results were significantly increased. Likewise, SRCNN helped to enhance the quality of low-resolution and motion blurring images; hence the retrieval accuracies were slightly increased. Besides, there was no metric overwhelming each other in performing retrieval.

V. Conclusions

This study aims to verify that features extracted from a CNN model can be used as information for image retrieval and whether the model trained on the fine dataset could also be applied for degraded samples. Through extensive experiments, the study has shown that convolutional layers learn features that specify common characteristics of objects in the same class. Those features compressed into feature vectors which maintain clustering assemble in the representation space. Experimental results also showed that the proposed framework is sensitive to noise and could not completely overcome by simply using enhancing techniques. However, to drive consistent conclusions on this topic, further research and experiments on different datasets are crucial.

References

[1] L. Yann, L. Bottou, Y. Bengio and H. Patrick, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* pp. 2278-2324, 1998.

[2] L. Yann, Y. Bengio and H. Geoffrey, "Deep learning," *nature,* vol. 521, no. 7553, pp. 436-444, 2015.

[3] Q. Tian, H. Li and Z. Wengang, "Recent advance in content-based image retrieval: A literature survey," *arXiv preprint arXiv:1706.06064,* 2017.

[4] L. Zheng, S. Wang, Z. Liu and Q. Tian, "Packing and padding: Coupled multi-index for accurate image retrieval," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014.

[5] A. Gordo, J. Almazan, J. Revaud and D. Larlus, "Deep image retrieval: Learning global representations for image search.," in *European conference on computer vision*, Cham, 2016.

[6] H. Noh, A. Araujo, J. Sim, T. Weyand and B. Han, "Large-scale image retrieval with attentive deep local features," in *Proceedings of the IEEE international conference on computer vision*, 2017.

[7] R. Filip, G. Tolias and O. Chum, "CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples," in *European conference on computer vision*, Cham, 2016.

[8] E. Cubuk, B. Zoph, D. Mane, V. Vasudevan and Q. V. Le, "Autoaugment: Learning augmentation strategies from data," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019.

[9] C. R. Gonzalez and E. R. Woods, Digital Image Processing 4th edition, Pearson, 2017.

[10] A. Nguyen, J. Yosinski and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

[11] Y. Lecun, H. J. Fu and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* vol. 2, 2004.

[12] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv preprint arXiv:1603.07285,* 2016.

[13] C. Dong, C. C. Loy, K. He and X. Tang, "Image super-resolution using deep convolutional networks.," *IEEE transactions on pattern analysis and machine intelligence,* vol. 38, no. 2, pp. 295-307, 2015.

Appendix. Additional Figures

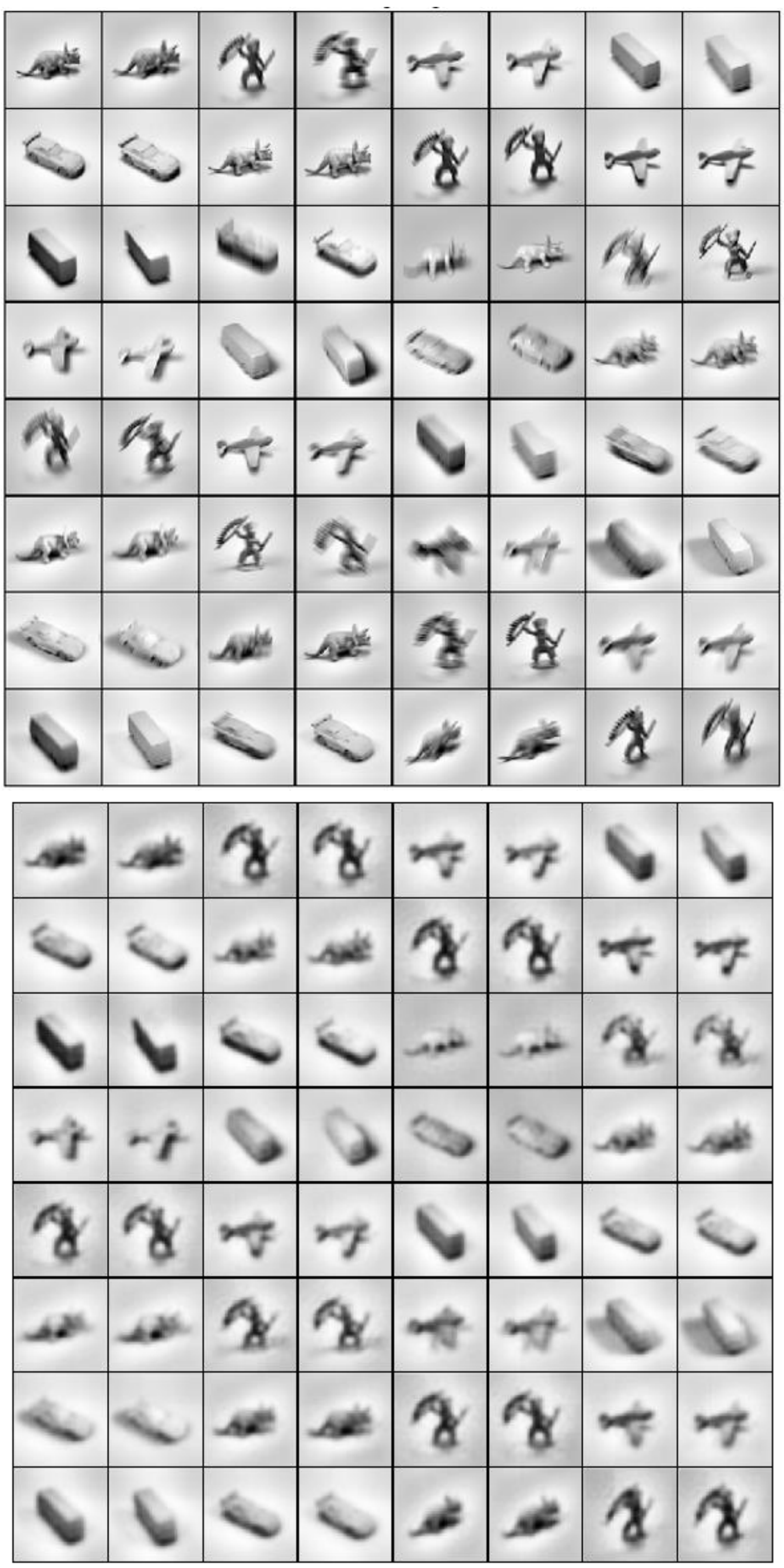
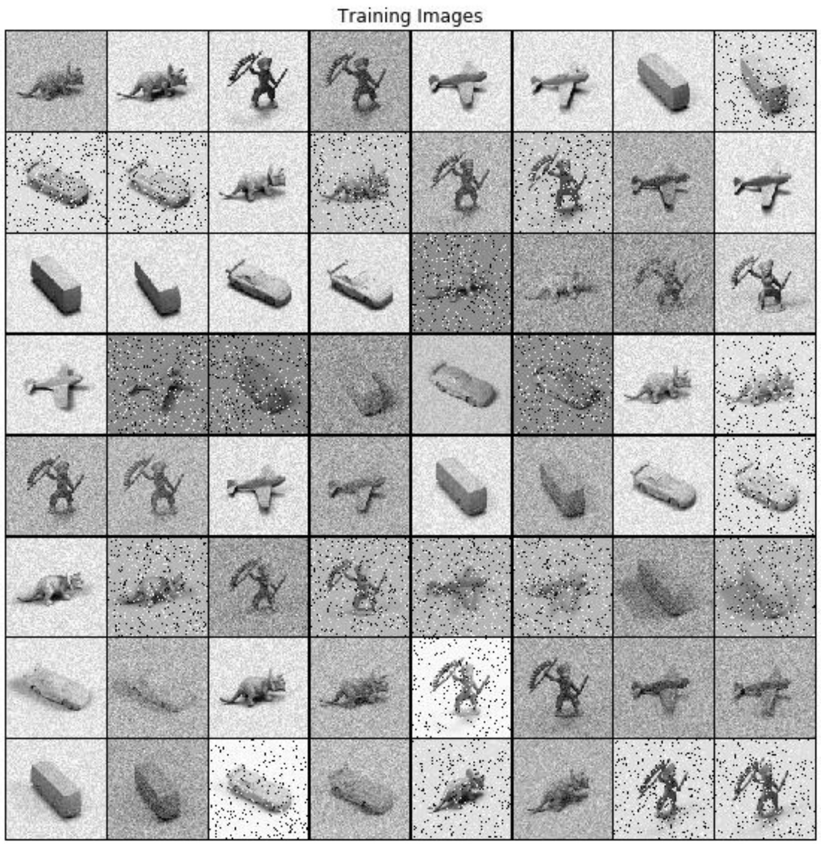


Figure A1. Motion blurring images and their enhancement



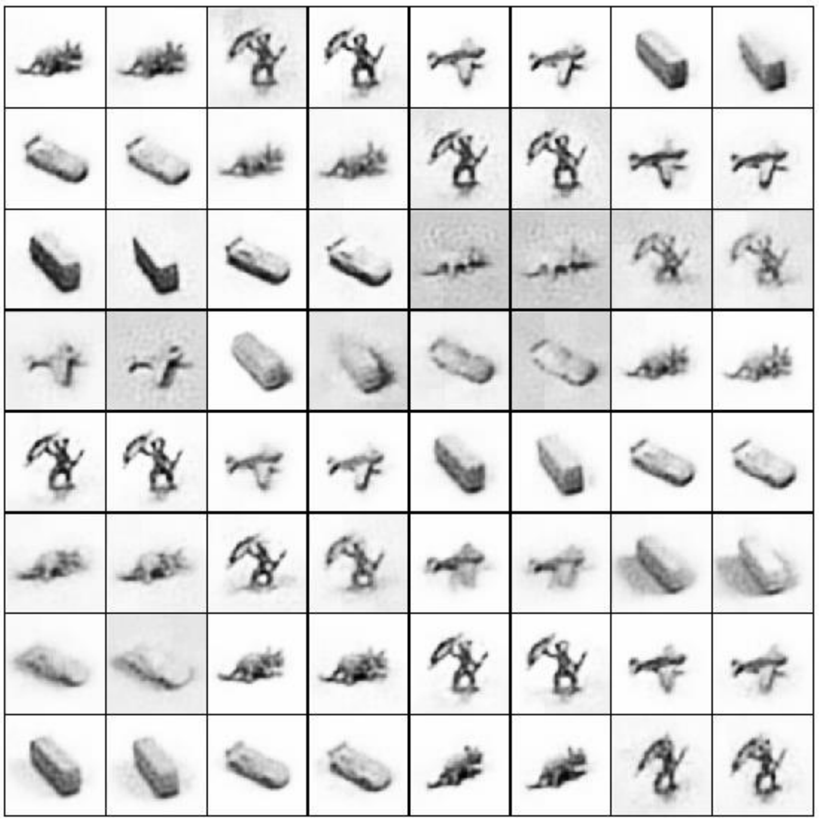


Figure A2. Noisy images and their enhancement

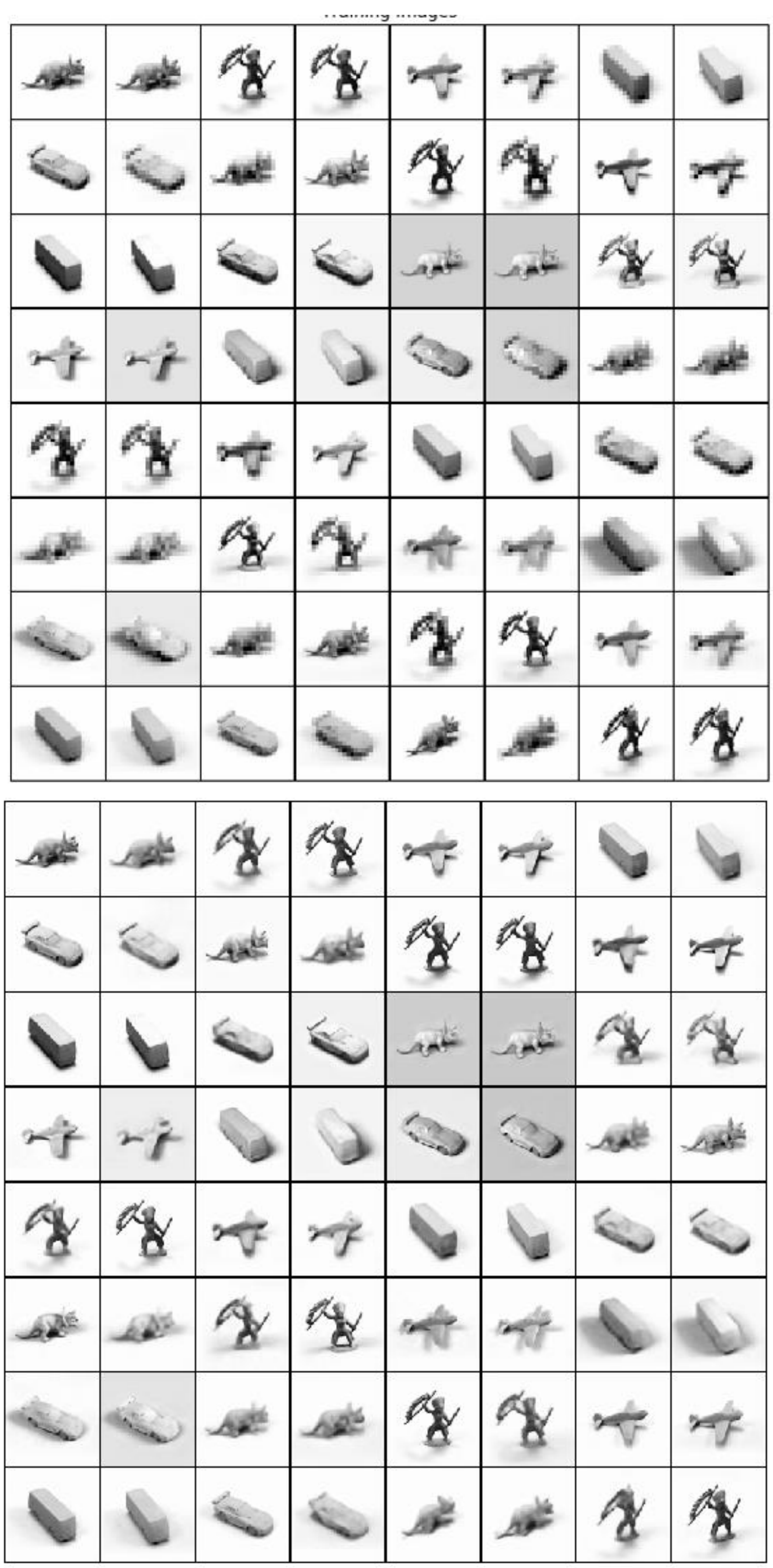


Figure A3. Low-resolution images and their enhancement

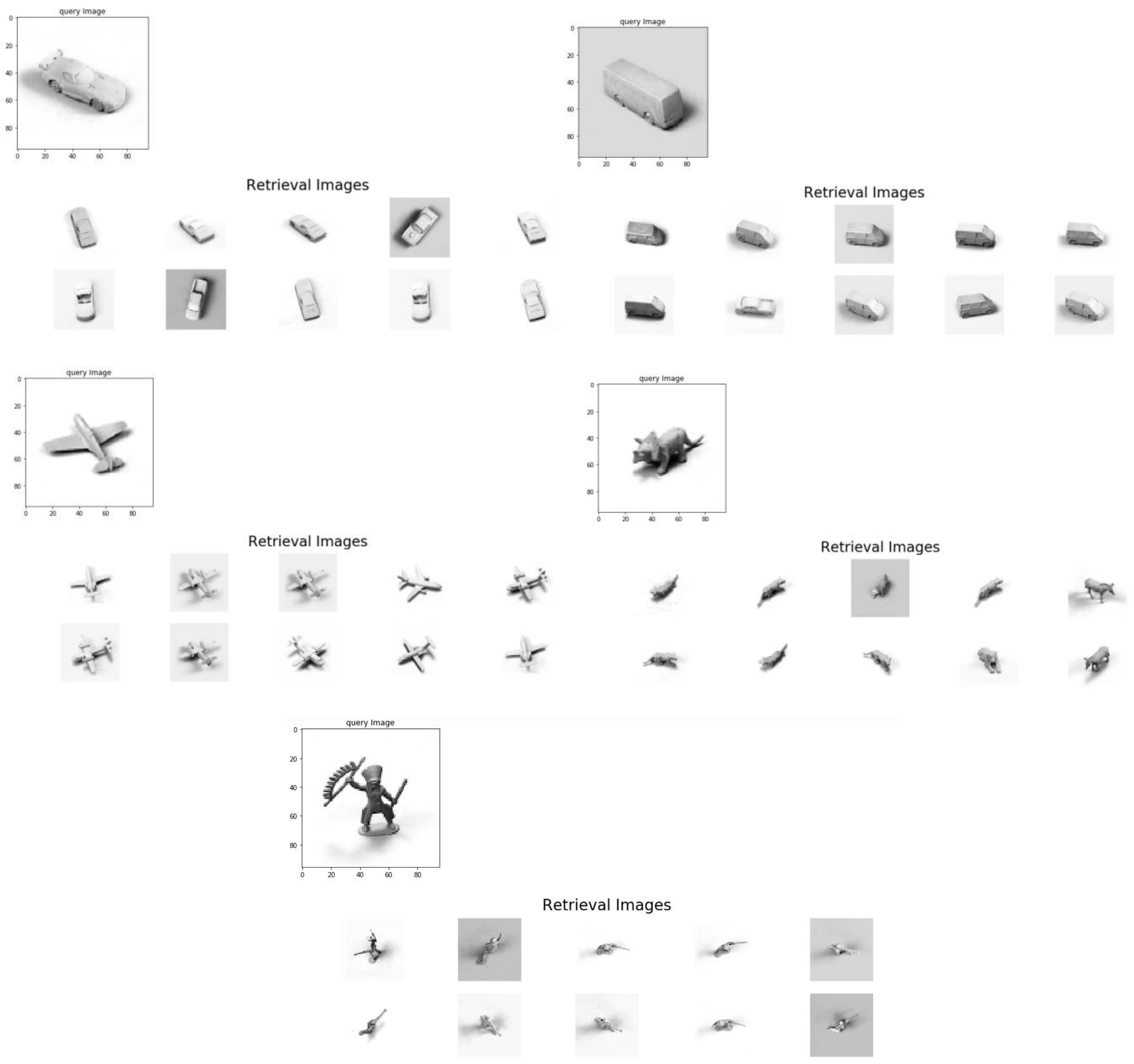


Figure A4. Retrieval results in the ideal case

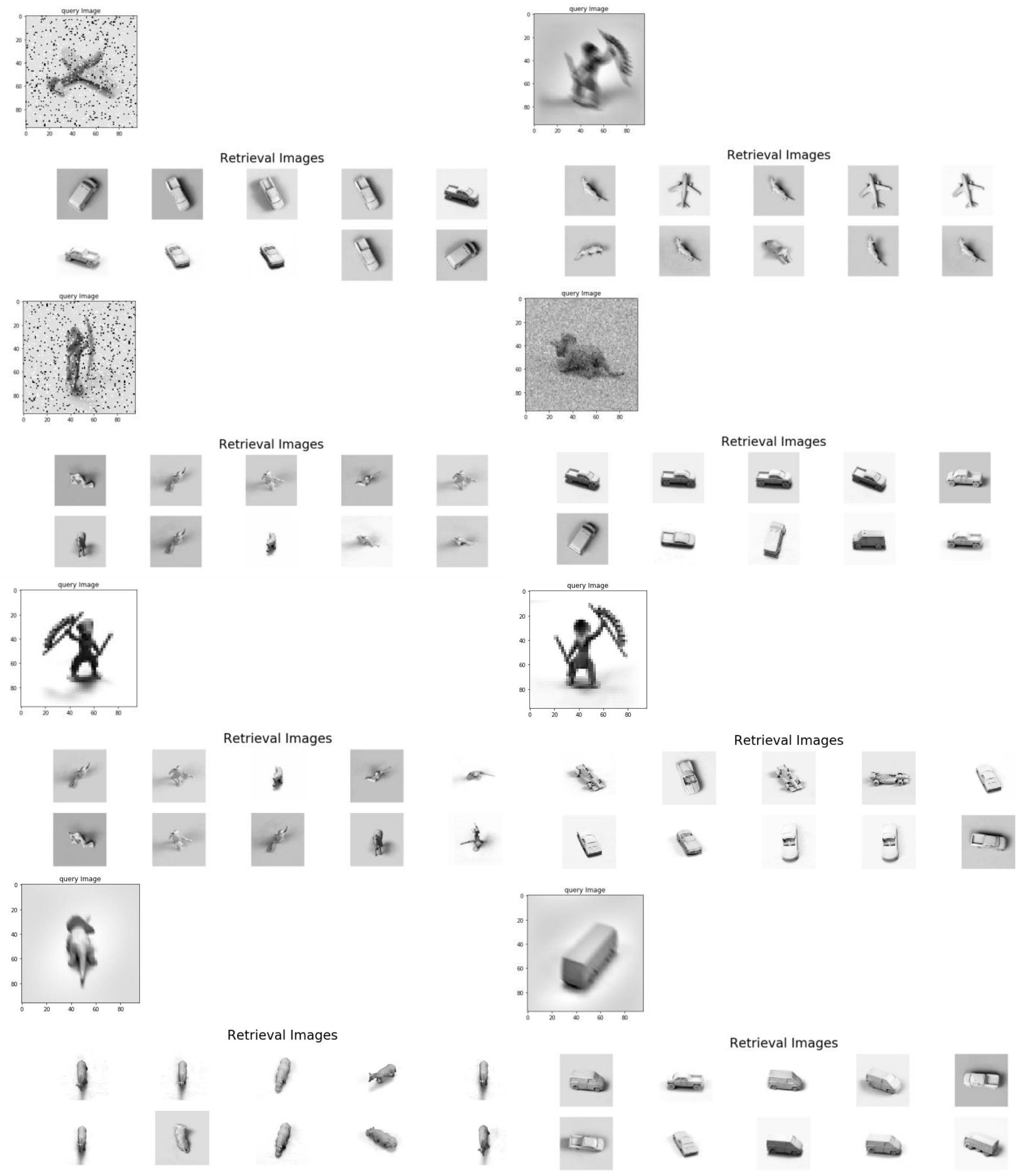


Figure A5. Retrieval results in degraded cases.