Bull. Soc. Photogr. Imag. Japan. (2024) Vol. 34 No. 2: 20–30

Original Paper

DNN Data Hiding for Transformer Encoder in ViT Models

Shuntaro FUKUOKA[a], Shoko IMAIZUMI[b], Minoru KURIBAYASHI[c]

Abstract: We investigated the effects of data hiding on the vision transformer (ViT), which is a transformer model, in this paper. In the field of deep neural network data-hiding, methods for protecting against piracy have been intensively studied for convolutional neural network (CNN) models. It has been observed that CNN models to which watermarking is applied show little effect in terms of model performance and training convergence. To the best of our knowledge, this is the first study to apply data hiding for ViT, which has a completely different architecture from the CNN model. We apply a quantization-based data hiding method to ViT and evaluate the effects on performance. Our experiments confirm that the proposed method does not cause ViT performance to degrade in terms of the classification accuracy and loss-function transition.

Key words: Vision transformer, data hiding, copyright protection, DM-QIM, loss function

1. Introduction

In recent years, more and more digital images, including photographs, have been exchanged over networks. In many cases, each image is labeled to separate it into categories and stored in a database in the cloud. Against this background, techniques for automatically classifying images using deep learning are rapidly developing. Deriving trained models requires massive amounts of computational resources and data, so it is essential to protect the trained models. Protection methods using data hiding for trained deep neural networks (DNNs) were actively studied in Refs. [1)–11)]. Zhang et al. [1)] proposed a method where a visible watermark is embedded into input images. In this method, marked images are classified under a different label. This leads to protecting the copyrights of model owners. In another approach, Zhao et al. [2)] proposed a method that embeds a watermark during a channel pruning process. Channel pruning reduces the size of DNNs by removing less significant weights. The method has high robustness through embedding into the percentage value of weights to be removed. Uchida et al. [3, 4)] protected DNNs by embedding watermarks into weights of the models. In this method, a watermark is embedded by multiplying a matrix derived from a secret key by the weights. The loss function caused by data hiding is then calculated and added to the training loss in model training. This method was further improved by extending its feature vector selection [5, 6)], but there exists a major issue where weights are significantly changed by data hiding. Another method detects an embedded watermark using a standard deviation of the weights and removes the watermark [7)]. Furthermore, Kuribayashi et al. [8, 9)] proposed a novel data hiding method using constant weight code (CWC) coding [10, 11)]. Their method is resistant to the pruning attack that removes redundant branches in the neural network and reduces the computational costs without seriously degrading the performance. The method also assumes that a non-fungible token (NFT) is used as the watermark to provide resistance against overwrite attacks in the method.

For the convolutional neural network (CNN), a method that embeds watermarks into weights of fully-connected layers has been proposed [12,13)]. The weights extracted from the fully-connected layers are first transformed into a frequency domain, and then the watermark is embedded into the frequency domain by using a discrete cosine transform (DCT). This operation diffuses the change in weights and thus suppresses the effects caused by data hiding. The confidentiality of the watermark can be enhanced by dither modulation in the process of data hiding/extraction. That method has used the dither modulation quantization index modulation (DM-QIM) method [14)] as an embedding method. The DM-QIM method is an embedding method based on the QIM method [15)]. The QIM method embeds a watermark into the frequency components by rounding them to the nearest value among the multiples defined as a step size. Here, the amount of change due to data hiding can be suppressed in the method, and it can be estimated by using the step size. In addition, the DM-QIM method introduces dither modulation to make it difficult for an attacker to identify the location where the data is embedded. Therefore, the use of the DM-QIM method for data hiding has less impact on the model than other embedding methods and is expected to improve the secrecy of the hidden data. The conventional method [12, 13)] is effective for data hiding to CNN. We extend the method to the vision transformer (ViT), which has been attracted as a current classification model with isotropic network,

Received 18th September, 2024; Accepted 11th November, 2024; Published 27th November, 2024
[a] Graduate School of Science and Engineering, Chiba University
1-33 Yayoicho, Inage-ku, Chiba-shi, Chiba 263-8522, Japan
[b] Graduate School of informatics, Chiba University
1-33 Yayoicho, Inage-ku, Chiba-shi, Chiba 263-8522, Japan
[c] Center for Data-driven Science and Artificial Intelligence, Tohoku University
41 Kawauchi, Aoba-ku, Sendai-shi, Miyagi, 980-8576, Japan

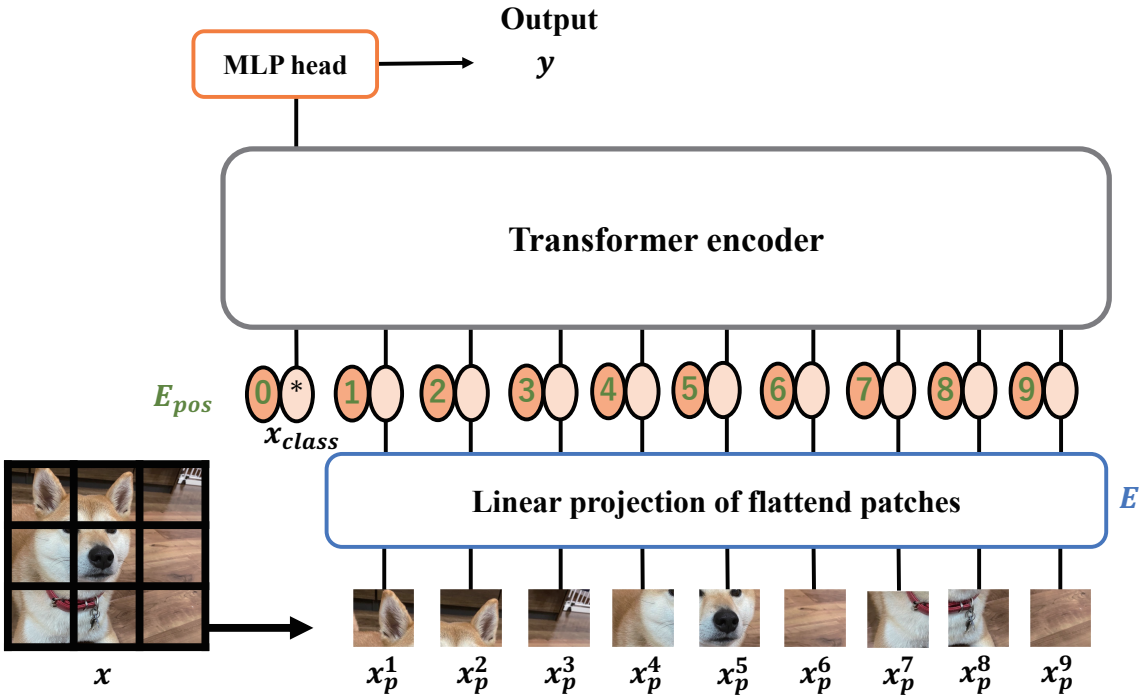


Fig. 1. ViT structure [16].

and explore its impact on ViT.

In this paper, we apply the DM-QIM method to embedding watermarks into ViT. ViT has recently attracted attention for its ability to attain a higher classification accuracy with less computational complexity than CNN. ViT introduces a transformer, which was originally used in natural language processing. Thus, the architecture of ViT differs from that of traditional image classification models. This difference may lead to conventional data hiding methods becoming incompatible with ViT and new attacks emerging. To accommodate the new architecture, we investigate a DNN data hiding method for ViT in this study. In the proposed method, a watermark is embedded in the multi-layer perceptron (MLP) and multi-head self-attention (MSA) for the reason that these operations are composed of a large number of weights. Focusing on the high redundancy of MLP and MSA in ViT, we propose a data hiding method that does not undermine ViT performance. First, we develop a method to effectively apply the DM-QIM method to ViT so as to suppress the amount of weight change due to data hiding. We then examine the effect of an increases in the number of weights affected by data hiding and the layers that are least affected by data hiding. Through our experiments, we prove that the degradation in accuracy is negligible even when the number of weights affected by data hiding increases. This is similar no matter which layer the watermark is embedded into.

2. Preparation

In this section, we briefly review ViT [16] and the DM-QIM method [14].

2.1 Vision Transformer (ViT)

In recent years, attention mechanisms have attracted much attention in the field of deep learning.

In natural language processing, a transformer with an attention mechanism enhanced the accuracy of machine translation [17]. Additionally, ViT, which is an image classification model, has achieved a higher classification accuracy than traditional models by incorporating a transformer [16].

Figures 1 and 2 show the fundamental structures of ViT and a transformer encoder, respectively. Initially, the input image $\boldsymbol{x} \in \mathbb{R}^{H\times W\times C}$ is divided into patches $\boldsymbol{x}_p^{\alpha} \in \mathbb{R}^{P^2C}$. Here, H, W, and C represent the height, width, and number of color channels of the input image, respectively. P denotes the patch size, and α indicates the patch number ($\alpha = 1, 2, \cdots, N$, where N is the total number of patches). Using the matrix $E \in \mathbb{R}^{P^2C\times D}$, each patch then undergoes a linear transformation to correspond to the input dimension D of the transformer encoder. A class token $\boldsymbol{x}_{class} \in \mathbb{R}^D$ is added to the linearly transformed patches. Subsequently, positional information $E_{pos} \in \mathbb{R}^{(N+1)\times D}$ is embedded into the linearly transformed patches and the class token, resulting in the input $\boldsymbol{z}_0 \in \mathbb{R}^{(N+1)\times D}$ to the transformer encoder. $\boldsymbol{z}_0$ is expressed as

$$\boldsymbol{z}_0 = (\boldsymbol{x}_{class}\, \boldsymbol{x}_P^1\boldsymbol{E}\, \boldsymbol{x}_P^2\boldsymbol{E} \cdots \boldsymbol{x}_P^1\boldsymbol{E})^T + \boldsymbol{E}_{pos}. \quad (1)$$

The transformer encoder consists of multiple layers. As shown in Fig. 2, each layer includes a multi-head self-attention (MSA), a multi-layer perceptron (MLP), and two layer normalizations (LNs). Here, let L and l denote the number of layers in the transformer encoder and the l-th layer index (l=0, 1, $\cdots$, L-1), respectively. First, $\boldsymbol{z}_l$ is converted to $LN1(\boldsymbol{z}_l)$ through the first layer normalization, i.e., LN 1. Next, $LN1(\boldsymbol{z}_l)$ is transformed into three vectors Q_i, K_i, and V_i ($i \in \{1, 2, \cdots, h\}$, where h denotes the number of heads) by multiplying it by trainable weight matrices $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d_{head}\times d_{head}}$, respectively. Note that d_{head} represents the number of dimensions of a head.

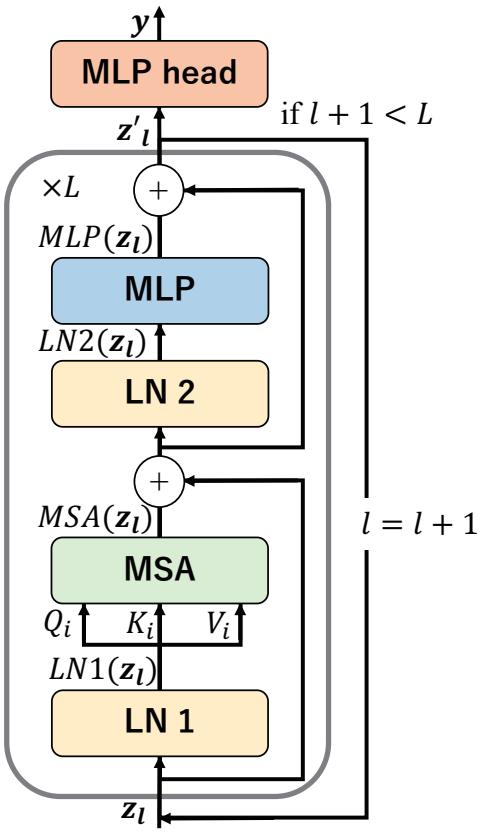


Fig. 2. Layered structure of transformer encoder [16].

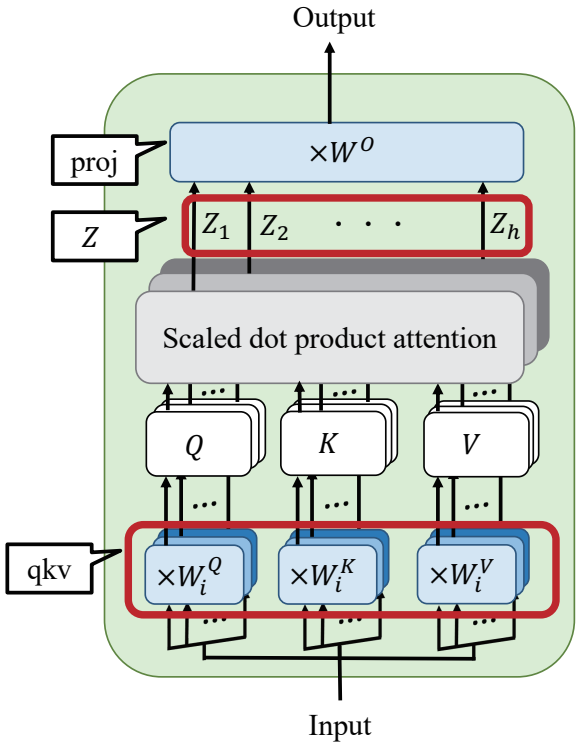


Fig. 3. Structure of multi-head self-attention (MSA) [18].

Figure 3 shows the structure of MSA. MSA has two types of learnable weights: qkv and proj. Note that these weights are referred to differently depending on the pre-trained model. qkv is a series of weights, which consists of W_i^Q, W^{iK}, W_i^V in Fig. 3. W_i^Q, W_i^K, W_i^V are multiplied by the input, and then the input is transformed into the three matrices Q, K, and V. proj is another series of weights, which corresponds to W^O in Fig. 3. proj is multiplied by $Z \in \{Z_1, Z_2, \cdots, Z_h\}$ to reduce the dimension of the matrix output from the scaled dot product attention. Attention values are expressed as

$$\boldsymbol{Attention}(\boldsymbol{Q_i}, \boldsymbol{K_i}, \boldsymbol{V_i}) = \boldsymbol{softmax}\left(\frac{\boldsymbol{Q_i K_i}^T}{\sqrt{\boldsymbol{d_{model}}}}\right)\boldsymbol{V_i}. \quad (2)$$

Now, we replace $Attention(Q_i, K_i, V_i)$ with $head_i$. By using the attention values $head_i$ and learnable weight matrix $W^O \in \mathbb{R}^{d_{head} \times d_{head}}$, $MSA(\boldsymbol{z}_l)$ is derived as

$$\boldsymbol{MSA}(\boldsymbol{z_l}) = \boldsymbol{Concat}(\boldsymbol{head_1}, \cdots, \boldsymbol{head_i}, \cdots, \boldsymbol{head_h})\boldsymbol{W^O}. \quad (3)$$

The result of adding $\boldsymbol{z}_l$ to $MSA(\boldsymbol{z}_l)$ is further normalized to $LN2(\boldsymbol{z}_l)$ through the second layer normalization, LN 2.

MLP derives $MLP(\boldsymbol{z}_l)$ from its input $LN2(\boldsymbol{z}_l)$. $\boldsymbol{z}'_l$ is the result of adding the input to LN 2 to $MLP(\boldsymbol{z}_l)$. In the case of l+1=L, $\boldsymbol{z}'_l$ should be the input to the MLP head. Otherwise, $\boldsymbol{z}'_l$ will be the input to the transformer encoder in the next layer, and the above process is repeated. The output of the MLP head is the final output $\boldsymbol{y}$. $\boldsymbol{y}$ is obtained by applying the following equation to the first row of $\boldsymbol{z}_{L-1}$, i.e., $\boldsymbol{z}^0_{L-1}$:

$$\boldsymbol{y} = \boldsymbol{LN}(\boldsymbol{z}^0_{L-1}). \quad (4)$$

2.2 Related Work

The conventional method [12, 13] embeds watermarks into the weights of fully-connected layers in CNNs using the DM-QIM method. To enhance the confidentiality of watermarked weights, this method uses a QIM method [15] and dither modulation (DM). We describe the embedding process below.

First, according to a secret key, n weights, where $n >> k$, are selected and transformed into the frequency domain. Let $\omega_j \in \{0, 1\}$, $(1 \le j \le k)$ be a watermark bit and $f \in \{f_1, f_2, \cdots, f_i, \cdots, f_n\}$ be the selected weight. We suppose that F_l is the frequency components transformed from f_i. k frequency components are then modified by the DM-QIM method [14] to embed a k-bit watermark. When F_j is quantized with a step size s, F_j is rounded to the nearest multiple of s. If $\omega_j = 0$, F_j is rounded to the nearest even multiple of s; otherwise, it is rounded to the nearest odd multiple of s. Hence, the quantized values should be included in the set $\{0, \pm s, \pm 2s, \pm 3s, \cdots\}$, which are integer multiples of s. In such a case, however, the watermarked frequency components can be easily identified by only observing frequency components. If an attacker attempts to identify the watermarked n weights, it is sufficient to check the frequency components to determine whether the values are discretized. Even if the step size s is unknown, the discretized frequency components indicate the presence of a hidden watermark.

The DM-QIM method carries out the DM by adding pseudo-random numbers before the QIM data hiding operation. A pseudorandom number generator derives r_j in the range $[-s/2, s/2]$. The watermarked frequency component $\bar{F}_j$ is then calculated:

$$\overline{F}_j = \text{DM-QIM}(F_j, \omega_j, s, r_j) = \begin{cases} s\left\lfloor \frac{F_j + r_j}{s} \right\rfloor - r_j & \text{if } \left\lfloor \frac{F_j + r_j}{s} \right\rfloor \bmod 2 = \omega_j \\ s\left\lfloor \frac{F_j + r_j}{s} + 1 \right\rfloor - r_j & \text{otherwise.} \end{cases} \quad (5)$$

where $\lfloor \cdot \rfloor$ is a floor function. In the extraction operation, ω_j is recovered followed by

$$\omega_j = \left\lfloor \frac{\overline{F}_j + r_j + \frac{s}{2}}{s} \right\rfloor \bmod 2. \quad (6)$$

Since it is difficult to detect ω_j unless r_j is disclosed, r_j serves in the role of a secret key in the DM-QIM method. In this method, the change caused by data hiding is spread over n weights. This makes it difficult for an attacker to add noise only to the k watermarked frequency components F_j without a secret key.

It has been also confirmed that the data hiding operation has a small effect on training convergence of the watermarked CNN.

3. Proposed Method

In this section, we propose a novel data hiding operation on ViT, which has an isotropic network and an attention mechanism instead of convolutions.

3.1 Overview

The change in weights caused by data hiding is generally the largest just after the process at the first epoch and converges as training progresses. In the conventional methods, the loss function caused by data hiding is combined with the loss function derived from training to reduce the effects of data hiding. In contrast, the proposed method updates the weights at the end of each epoch in fine-tuning without using the loss function caused by data hiding. Such an operation can be interpreted as restricting the weights selected for data hiding to satisfy a specific condition provided by the watermark and secret key in the proposed method. The ViT parameters are updated at each epoch to converge into a local minimum. If the number of selected weights is small, the convergence speed is not seriously affected by the data hiding operation.

The amount of change in the embedded target before and after QIM can be estimated statistically and has been confirmed to be smaller than that of other data hiding methods [15]. Therefore, it is expected that the effects caused by the QIM method are also small. To conceal the embedding position, each single bit is not directly embedded into the individual weights of layers in the transformer encoder but rather embedded into the frequency domain after DCT. In the proposed method, we suppose that ViT has L layers and embed a watermark into α layers out of all layers. First, n weights are randomly sampled from α layers using a secret key, and the sampled weights are transformed into frequency domain by DCT. Next, to spread out the changes, k values are selected from the n frequency components. The data hiding operation and the amount of weight change caused by data hiding are described in detail below.

3.2 Data Hiding

In the proposed method, we embed a watermark by using the DM-QIM method for copyright protection of the trained ViT. The watermark is embedded into the weights of a transformer encoder through fine-tuning. The detailed procedure is as follows.

Step 1. Sample n weights from α layers in the transformer encoder according to a secret key. A series of the weights, f, is represented as:

$$f = (f_1, f_2, \ldots, f_n) \quad (7)$$

Step 2. Derive the frequency components F by full-domain DCT on f:

$$F = \text{DCT}(f) \quad (8)$$

Step 3. k-bit watermark $\omega = (\omega_1, \ldots, \omega_j, \ldots, \omega_k)$ is embedded into the DCT coefficients F_j:

$$\overline{F}_j = \begin{cases} \text{DM-QIM}(F_j, \omega_j, s, r_j) & 1 \le j \le k \\ F_j & j > k, \end{cases} \quad (9)$$

where $\overline{F}$ is the watermarked DCT coefficients.

Step 4. The watermarked weights $\overline{f}$ are obtained by inverse DCT:

$$\overline{f} = \text{IDCT}(\overline{F}) \quad (10)$$

Note that we can theoretically embed an equivalent number of bits to the number of weights. The extraction procedure is the same as the data hiding procedure, and each watermark bit ω_j is extracted by using Eq. (6).

In the proposed method, even when an attacker attempts to overwrite the watermark, it is difficult for the attacker to remove the previous watermark without a secret key. However, it is possible for the attacker to embed another watermark using a different key. In such a case, there should exist two kinds of watermarks in ViT in random order so that it is difficult to distinguish the original from the malicious. The same as the conventional method [12,13], we introduce a non-fungible token (NFT) to assure that there is an operation history of data hiding.

3.3 Target Layers of Data Hiding

As target layers for embedding watermarks, we focus on the MLP and MSA layers in a transformer encoder in ViT. These layers possess a large amount of redundancy, so it is expected that the effects of data hiding can be suppressed.

1) MLP: MLP is a feed forward neural network. It has an input layer that consists of neurons as receivers, one or more hidden layers that perform computations and undergo iterations, and an output layer that predicts the final output. During the propagation of values from one layer to another, the weights are multiplied by the propagated values. These multiplied weights are updated by training. There are more than 500,000 weights in a single MLP layer of ViT. Owing to such redundancy, a watermark can be embedded into the MLP layer without seriously degrading the ViT performance.

It is known that DNN has many local minima and almost all the local minima are extremely similar to the global minimum [19, 20]. Even when small subsets of weights are slightly changed, the effects can be reduced by adjusting the remaining weights.

2) MSA: MSA has two types of weights: qkv and proj. qkv is a weight matrix that is multiplied by the input to transform it into three vectors, Q, K and V. proj is a weight matrix for reducing the dimension of matrix, which is generated by scaled dot product attentions. qkv comprises approximately 500,000 weights, while proj consists of around 1,000 weights. In this paper, we embed the watermark into qkv, where the number of weights is much larger than n, to avoid causing significant changes.

3.4 Amount of Change in Weights by Data Hiding

The change in weights caused by data hiding can be estimated. Since the weights in the transformer encoder are balanced, there exist no singular values in sampled weights f. This helps avoid deriving bias in F caused by transforming f into the frequency domain. Therefore, we can expect that the amount of change in weights caused by data hiding uniformly varies in the range of $[-s, s]$, where s is a step size. In embedding a single bit into each F, the expected change ϵ is calculated by

$$\epsilon = \frac{1}{2s}\int_{-s}^{s} t^2 dt = \frac{s^2}{3}. \tag{11}$$

In the case of embedding k bits, the total change E_ω is estimated by

$$E_\omega = \frac{ks^2}{3}. \tag{12}$$

After performing the inverse DCT, E_ω is distributed over n weights. As a result, the expected change for each single weight is E_ω/n.

Additionally, for each epoch, a watermark is embedded into the selected n weights. The n weights are updated by training in common with other weights. To prevent losing the watermark, the same watermark is repeatedly embedded into these weights at each epoch. This data hiding process may increase the convergence time in the training and degrade the final-model performance. In our method, however, the effects of data hiding are negligible because the number of the selected weights n is significantly small relative to the total number of weights. In addition, since the expected change in each single weight is expressed as E_ω/n, we expect that the increase in n reduces the amount of weight change and contributes to suppressing the degradation of the model performance. We confirm this in the next section.

4. Experimental Results

In this section, we evaluate the effects of embedding a watermark into ViT with the DM-QIM method. We first describe the experimental conditions and then discuss the effects of the number of selected weights n and the layer where the watermark is embedded in terms of the classification accuracy and loss function. Note that all the watermark bits are correctly extracted in our method. Since the results may fluctuate following the default settings, we performed the experiment five times and calculated their mean. In this paper, we embed the watermark into the MLP and qkv of MSA.

4.1 Experimental Setup

We use a pre-trained model called *vit_small_patch16_224* [17] from timm, which is a PyTorch Image Models library. *vit_small_patch16_224* was trained using over 1 million images from the ImageNet database [21] and over 300 million images from a dataset called JFT-300M provided by Google. In this experiment, we embedded the watermark into the MLP or qkv of MSA in all 12 layers. Data hiding and fine-tuning were performed on this pre-trained model while obtaining the classification accuracy and loss function at each epoch. We summarize the hyperparameters for fine-tuning in Table 1. These values were also used for the previous method [9], and we used them in our experiments. In fine-tuning, we used the Dogs vs. Cats image dataset, including 25,000 images for training and 12,500 images for validation, from the Kaggle database[1]. We assumed NFT class tokens as the embedded watermark, so the watermark amount should be 256 bits. To investigate the sensitivity of layers to be watermarked, we set $\alpha = 1$ and explored the performance degradation for each layer. The number of parameters in the trained model was approximately 87 million. We sampled n values from these 87 million weights using a secret key.

4.2 Experimental Results

We evaluated the classification accuracy and loss function. The number of weights n was configured to 2,048. The weights were initially extracted from the MLP or qkv of MSA, and the watermark was then embedded into the weights. The weights were finally trained. We defined a series of the above processes as one epoch. For the proposed method, we assumed an NFT as the embedded watermark, so the watermark length was 256 bits ($k = 256$). Additionally, the quantization step size s was set to 8 for the DM-QIM method.

4.2.1 Number of Selected Weights

Here, we evaluated the effects caused by the number of selected weights n. We changed n from 2,048 to 4,096, 6,144, and 8,192. Note that we embedded the watermark into the weights of the first layer (L=0) of the MLP. Figure 4 shows the transitions in classification accuracy and loss function during training with data hiding. The transitions were similar to those observed during the training without data hiding. From these results, we can say that the number of n hardly affected the ViT performance when a 256-bit watermark was embedded. In 3.4, we estimated that the performance degradation of the model would be suppressed as n was increased. This is because it was expected that fewer weights would be changed by data hiding as n became larger. However, the classification accuracy and convergence were comparable for any value of n in our experiment.

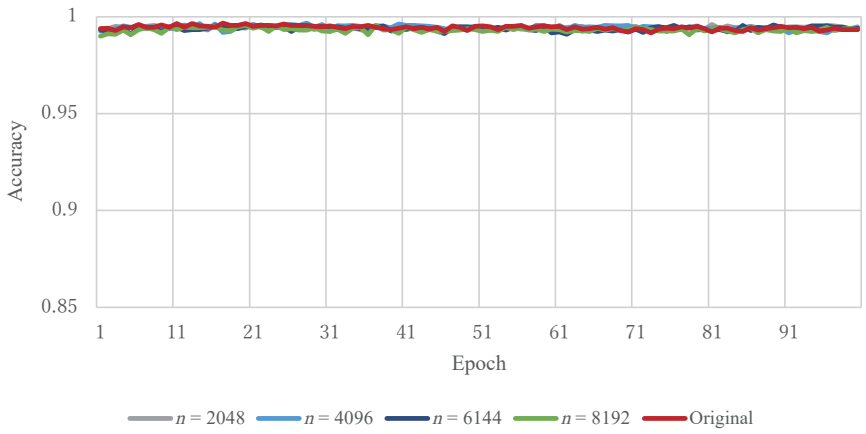
4.2.2 Target Layer for Data Hiding

As mentioned above, data hiding was applied to the MLP or qkv of MSA in one of the 12 layers to investigate its effects.

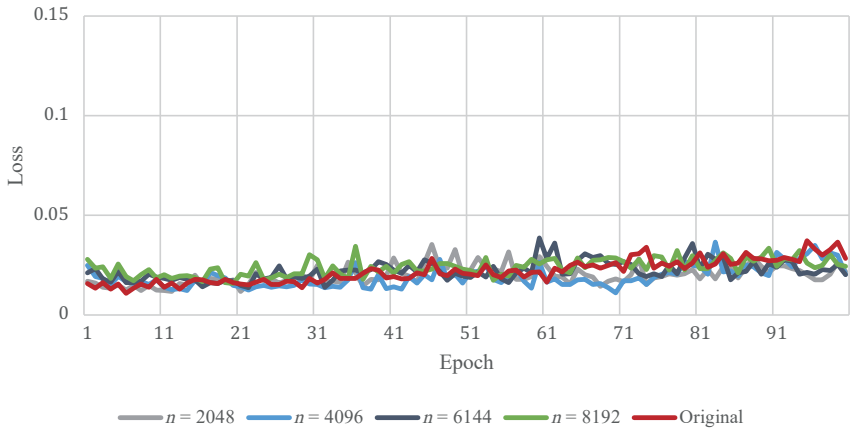
1) Multi-Layer Perceptron (MLP):

Figure 5 depicts the transitions in the classification accuracy and loss function at each epoch. The transitions with data hiding had

1 https://www.kaggle.com/datasets/karakaggle/kaggle-cat-vs-dog-dataset

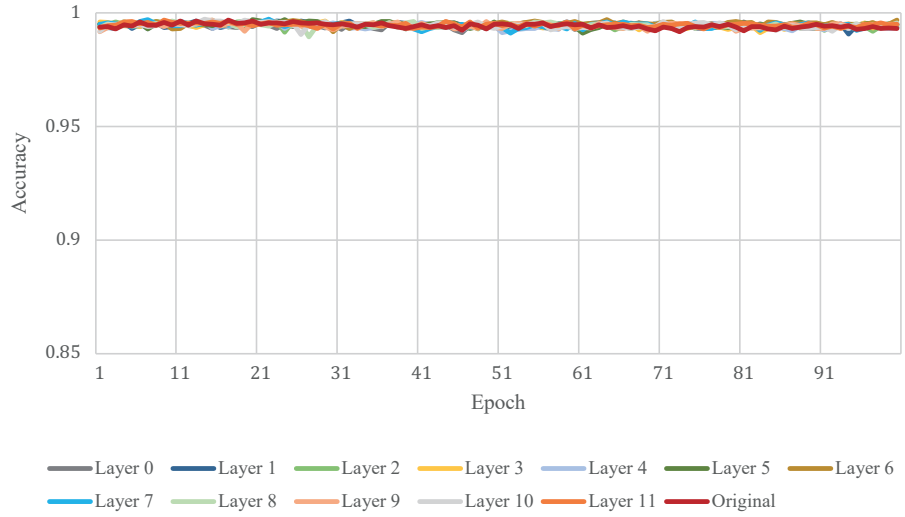


(a) Classification accuracy

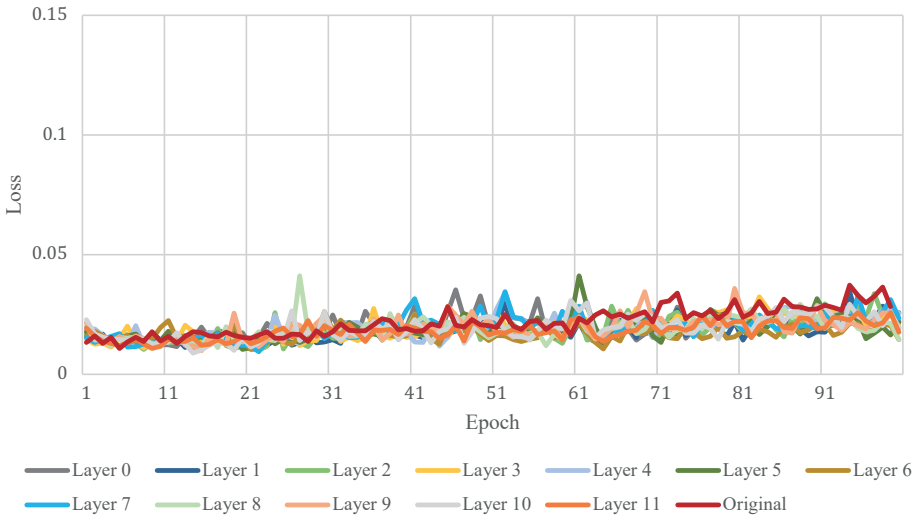


(b) Loss function

Fig. 4. Transition in performance with progress of training process (increasing n).



(a) Classification accuracy

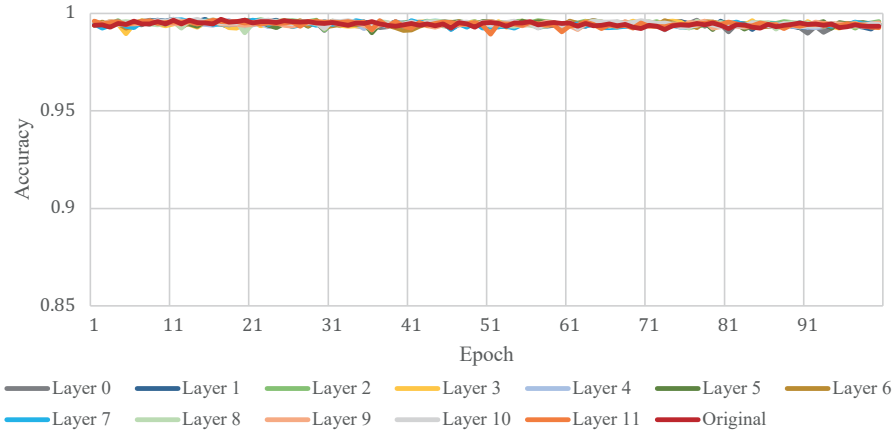


(b) Loss function

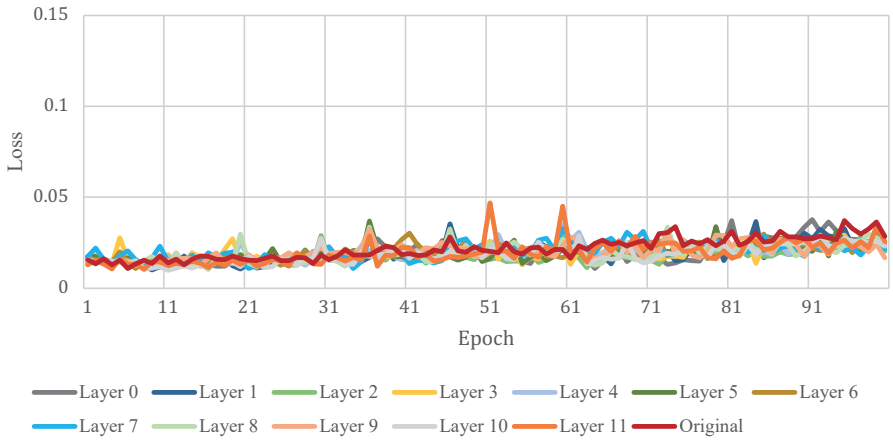
Fig. 5. Transition in performance with progress of training process (MLP).

TABLE 1. Hyperparameters for fine-tuning.

Patch size	16
Batch size	64
Learning late	0.00003
# of epochs	100

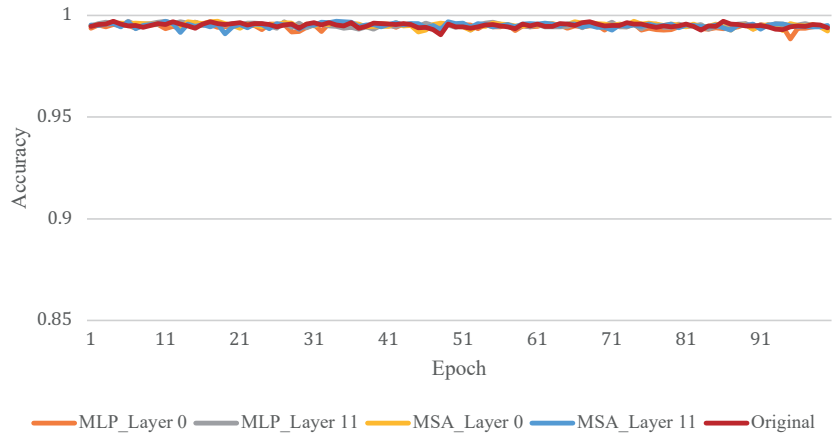


(a) Classification accuracy

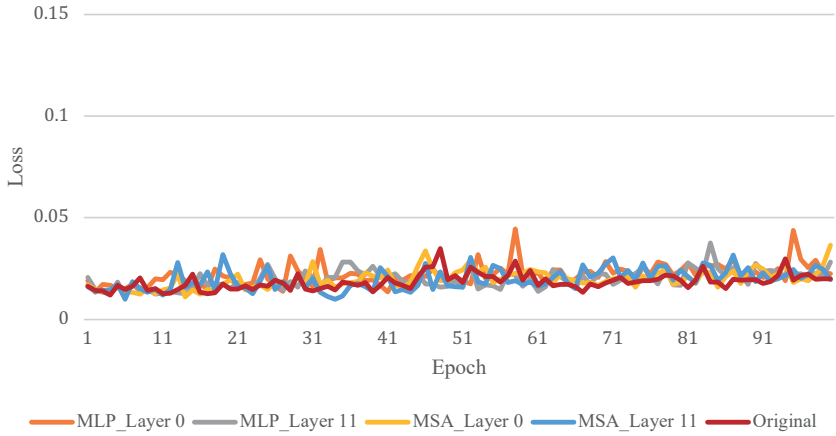


(b) Loss function

Fig. 6: Transition in performance with progress of training process (MSA).

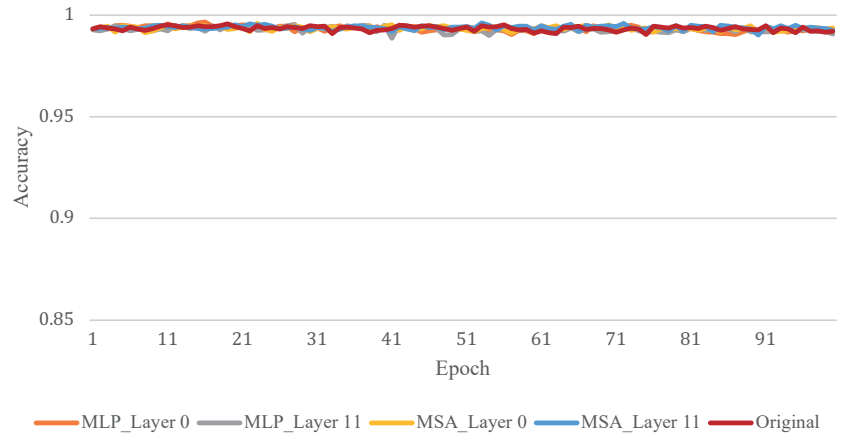


(a) Classification accuracy

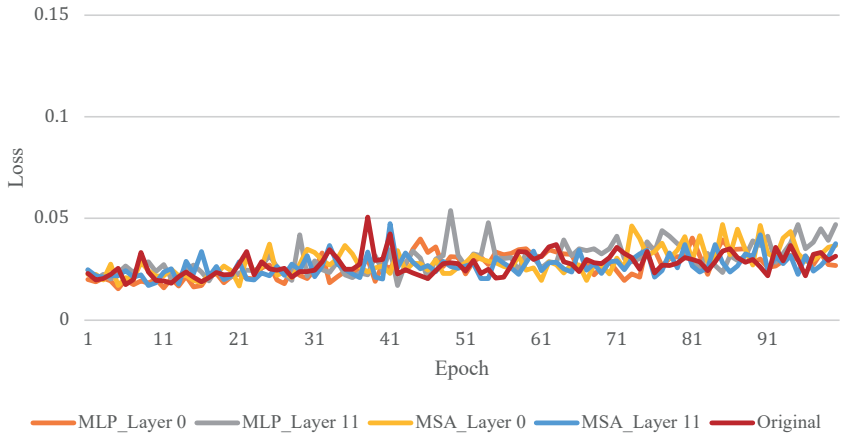


(b) Loss function

Fig. 7: Transition in performance with progress of training process for *vit_small_patch16_224.augreg_21k*.



(a) Classification accuracy



(b) Loss function

Fig. 8: Transition in performance with progress of training process for *vit_base_patch16_224.augreg_1k*.

similar shapes to those without data hiding. Training with data hiding for the CNN caused both the classification accuracy to degrade and the loss function to increase in the early stage of the training. In contrast, it is clearly seen that training with data hiding for ViT never had such effects through all epochs. Consequently, in the case that a 256-bit watermark is embedded into the MLP of any layer, the ViT performance can be preserved.

2) Multi-Head Self-Attention (MSA):

Figure 6 shows the transitions in the classification accuracy and loss function during training with data hiding. These transitions were analogous to those observed during training without data hiding. As is the case with MLP, the performance is similar when a 256-bit watermark is embedded into the qkv of MSA of any layer.

4.2.3 Other Models

Here, we applied the DM-QIM method to different ViT models and confirmed that the results are equivalent to the model of *vit_small_patch16_224*. As the different pre-trained models, we used *vit_small_patch16_224.augreg_21k* and *vit_base_patch16_224.augreg_in1k*. First, *vit_small_patch16_224.augreg_21k* is a model pre-trained using ImageNet-21k, which is an image dataset with more than 21,000 classes. Thus, this model works for a greater variety of classes than *vit_small_patch16_224*. On the other hand, *vit_base_patch16_224.augreg_in1k* pre-trained using ImageNet has more than twice as many parameters than *vit_small_patch16_224*. This contributes to a higher accuracy than *vit_small_patch16_224*.

For the two models, we show the transitions in the classification accuracy and loss function at each epoch in Figs. 7 and 8. A watermark was embedded into one of the layers from Layer 1 and Layer 11 of the MSA and MLP. These transitions are analogous to those observed during training without data hiding. Consequently, the transitions show similar trends regardless of the model in which the watermark is embedded. In other words, the proposed method can embed a watermark while maintaining the ViT performance even when the number of weights or classes in a pre-trained model increases. Additionally, the proposed method can completely extract the watermark as well as the conventional method [12, 13].

4.4 Considerations

We first discuss the reason why the increase in n had no effect on model performance. It is observed from Fig. 4 that the number of weight parameters to be embedded has no effect on model performance. Regardless of the amount of weight change E_ω/n at the first epoch, the degradation in model performance is negligibly small

because the number of weights changed by data hiding is considerably smaller compared with the entire number of weights. In the training phase, the weights are updated at each epoch to approach a local minima, while the embedding operation changes the updated weights to move away from the local minima. However, the change caused by the embedding operation at each epoch is quite small and does not affect the model convergence in the training phase. We infer that this leads to accuracy preservation against changes due to data hiding.

We now discuss the sensitivities to external noise at each layer in which the watermark is embedded. The experimental results show that data hiding into a layer with a large number of weights, i.e., MLP or qkv of MSA, neither degrades classification accuracy nor increases the output of the loss function. On the other hand, there was no significant difference in performance when embedded in any of the L layers, where L=12 in this experiment. Thus, we can conclude that the layer position can be selected arbitrarily.

As we described, the proposed method is a novel method of data hiding into an arbitrary layer of either MLP or MSA in ViT with little impact on the model performance and convergence at the training model. There exist two concerns in ViT in applying the proposed method. First, it is difficult to embed a watermark into the normalization layers of ViT with a small number of weights. This is due to less weights in each normalization layer, (e.g., in *vit_small_patch16_224*, there are only 384 weights in each normalization layer); the number is too small to embed a watermark. As its second concern, an embedded watermark may not be extracted if the model encounters compression using channel pruning. The model compression through channel pruning changes the weights to reduce the model size. This weight change could remove the watermark. As the countermeasure to channel pruning, the use of constant weight code has been studied in Refs. [8,9]. It is one of promising methods to make it robust against channel pruning and one of our future works.

5. Conclusion

In this paper, we evaluated the effects caused by data hiding using the DM-QIM method on the classification accuracy and loss function in ViT. We embedded a watermark into the MLP or qkv of MSA in one of all layers of the transformer encoder. Both the accuracy and loss of the watermarked model converged similarly to the original model. Additionally, the performance could be preserved after data hiding without any notable degradation. The experimental results proved that we could select weights for data hiding from the MLP and qkv of MSA in an arbitrary layer.

References

1) J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. Molloy, "Protecting intellectual property of deep neural networks with watermarking," in Proc. ACM Asia Conference on Computer and Communications Security, pp. 159-172, 2018.
2) X. Zhao, Y. Yao, H. Wu, and X. Zhang, "Structural watermarking to deep neural networks via network channel pruning," in Proc. IEEE International Workshop on Information Forensics and Security, pp. 1-6, 2021.
3) Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding watermarks into deep neural networks," in Proc. ACM International Conference on Multimedia Retrieval, pp. 269-277, 2017.
4) Y. Nagai, Y. Uchida, S. Sakazawa, and S. Satoh, "Digital watermarking for deep neural networks," Int. J. Multimed. Inf. Retr., vol. 7, pp. 3-16, 2018.
5) B. D. Rouhani, H. Chen, and F. Koushanfar, "DeepSigns: An end-to-end watermarking framework for ownership protection of deep neural networks," in Proc. ACM International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 485-497, 2019.
6) H. Chen, B. D. Rouhani, C. Fu, J. Zhao, and F. Koushanfar, "DeepMarks: A secure fingerprinting framework for digital rights management of deep learning models," in Proc. ACM International Conference on Multimedia Retrieval, pp. 105-113, 2019.
7) T. Wang and F. Kerschbaum, "Attacks on digital watermarks for deep neural networks," in Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 2622-2626, 2019.
8) M. Kuribayashi, T. Yasui, and A. Malik, "White box watermarking for convolution layers in fine-tuning model using the constant weight code," J. Imaging, vol. 9, no. 6, 2023.
9) T. Yasui, T. Tanaka, A. Malik, M. Kuribayashi, "Coded DNN watermark: robustness against pruning models using constant weight code," J. Imaging, vol.8, no.6, 2022.
10) J. P. M. Schalkwijk, "An algorithm for source coding," IEEE Trans. Inf. Theory, vol. 18, no. 3, pp. 395-399, 1972.
11) A. E. Brouwer, J. B. Shearer, N. J. A. Sloane, and W. Smith, "A new table of constant weight codes," IEEE Trans. Inf. Theory, vol. 36, no. 6, pp. 1334-1380, 1990.
12) M. Kuribayashi, T. Tanaka, and N. Funabiki, "Deepwatermark: Embedding watermark into dnn model," in Proc. APSIPA Annual Summit and Conference, pp. 1340-1346, 2020.
13) M. Kuribayashi, T. Tanaka, S. Suzuki, T. Yasui, and N. Funabiki, "White-box watermarking scheme for fully-connected layers in fine-tuning model," in Proc. ACM Workshop on Information Hiding, Multimedia and Security, pp. 165-170, 2021.
14) M. Kuribayashi, T. Fukushima, and N. Funabiki, "Robust and secure data hiding for PDF text document," IEICE Trans. Inf. Syst., vol. E102.D, no. 1, pp. 41-47, 2019.
15) B. Chen and G.W. Wornel, "Quantization index modulation: a class of provably good methods for digital watermarking and information embedding," IEEE Trans. Inf. Theory, vol. 47, no. 4, pp. 1423-1443, 2001.
16) A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in Proc. International Conference on Learning Representations, pp. 1-21, 2021.
17) A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in Proc. Advances in Neural Information Processing Systems, pp. 5998-6008, 2017.
18) K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu, Z. Yang, Y. Zhang, and D. Tao, "A survey on vision transformer," IEEE Trans. Pattern Anal. Mach. Intell., vol. 45, pp. 87-110, 2023.
19) Y. N. Dauphin, R. Pascanu, C¸ Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization," in Proc. Advances in Neural Information Processing Systems, pp. 2933–2941, 2014.
20) A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun, "The loss surfaces of multilayer networks, " in Proc. International Conference on Artificial Intelligence and Statistics, pp. 192-204, 2015.
21) J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 248-255, 2009.