Learning to Find Correlated Features by Maximizing Information Flow in Convolutional Neural Networks

Wei Shen  Fei Li  Rujie Liu
Fujitsu Research & Development Center, Beijing, China.
{shenwei, lifei, rjliu}@cn.fujitsu.com

Abstract

Training convolutional neural networks for image classification tasks usually causes information loss. Although most of the time the information is redundant with respect to the target task, there are still cases where discriminative information is also discarded. For example, if images that belong to the same category have multiple correlated features, the model may only learn a subset of the features and ignore the rest. This may not be a problem unless the classification in the test set highly relies on the ignored features. We argue that the discard of the correlated discriminative information is partially caused by the fact that the minimization of the classification loss doesn’t ensure to learn all discriminative information but only the most discriminative information given the training set. To address this problem, we propose an information flow maximization (IFM) loss as a regularization term to find the discriminative correlated features. With less information loss the classifier can make predictions based on more informative features. We validate our method on the shiftedMNIST dataset and show the effectiveness of IFM loss in learning representative and discriminative features.

1. Introduction

Usually a classification model is trained with a softmax loss which is quite successful in many scenarios. This loss typical helps the model learn discriminative features for the target task and ignore the irrelevant features. However, if there are several discriminative features that are correlated within a category, the model may choose the most discriminative feature (e.g. color and texture) and ignore the rest (e.g. object structure). The reason is that the most discriminative features are those that make the steepest descend in loss function. As the training continues, those features will dominate the final feature representation and the rest discriminative features will be discarded as well as the irrelevant features. Similar evidence can also be found in recent study on ImageNet-trained CNNs. It shows that those models are biased towards texture rather than object shape [5]. Learning partial discriminative features does not make the most of the training dataset and thus it reduces the generalization capability of the model.

Information theory has been widely used to improve the representation capability of deep neural networks [2, 4, 6, 11, 12, 14]. In this work, we focus on how to apply mutual information to find correlated features in image classification tasks. According to Data Processing Inequalities (DPI), the mutual information between the input data and the hidden layers are decreasing as the layer goes deeper [13]. The main idea of information bottleneck (IB) trade off is that we can try to minimize the mutual information between the input data and the hidden representation and maximize the mutual information between the hidden representation and the label to find the optimal achievable representations of the input data [13]. In contrast, we find that when discriminative features are correlated, the maximization instead of minimization of mutual information between hidden representations can provide more information to the classifier for learning discriminative features. We call this strategy information flow maximization (IFM) which is achieved by estimating and maximizing the mutual information between convolutional layers simultaneously. IFM is implemented using a multi-layer fully connected neural network and it serves as a plug-in in the conventional CNNs in the training stage. In the test stage, all IFM blocks are removed and thus there is no extra computation cost.

2. Related work

There are many work concentrating on information maximization for deep networks. In [2], Chen et al. introduce InfoGAN which is a generative adversarial network that maximizes the mutual information between a small subset of the latent variables and the observation. In [1], Belghazi et al. present a Mutual Information Neural Estimator (MINE) that estimates mutual information between high dimensional continuous random variables by gradient descent.
over neural networks. In [6], Hjelm et al. introduce Deep InfoMax (DIM) to maximize mutual information between a representation and the output of a deep neural network encoder to improve the representation’s suitability for down-stream tasks. In [9], Jacobsen et al. propose a flow-based invertible network architecture and an alternative objective that extract overall discriminative knowledge in the prediction model.

The difference between our work and [6] is that we are concentrating on maximizing the mutual information between adjacent layers so that the information loss between the input data and their final representation can be reduced while [6] tries to maximize the mutual information between the final representation and the output convolutional feature maps. The work in [9] is closely related to our work, the main difference is that we apply IFM blocks instead of flow-maps. The work in [9] is closely related to our work, the main difference is that we apply IFM blocks instead of flow-maps. The work in [9] is closely related to our work, the main difference is that we apply IFM blocks instead of flow-maps. The work in [9] is closely related to our work, the main difference is that we apply IFM blocks instead of flow-maps. The work in [9] is closely related to our work, the main difference is that we apply IFM blocks instead of flow-maps.

3. Method

The pipeline of the proposed method is shown in Figure 1. The backbone network is a vanilla convolutional neural network. The IFM blocks are plugged in between adjacent convolution layers. Note that the IFM blocks are only used in the training stage. In the test stage, those IFM blocks are removed so that there is no extra computation cost.

![Figure 1. The training pipeline of the proposed method. In the training stage, the model is trained to minimize the classification loss and maximize the information flow between layers.](image)

3.1. Mutual information estimation

In order to be self-contained, in this section we will introduce how to estimate mutual information. Formally, the mutual information is calculated as

\[
I(X, Z) = \sum_{z \in Z} \sum_{x \in X} p(x, z) \log \frac{p(x, z)}{p(x)p(z)}
\]

\[
= KL(p(x, z)||p(x)p(z)),
\]

where \(X\) and \(Z\) are two random variables. \(p(x, z)\) is the joint probability function of \(X\) and \(Z\). \(p(x)\) and \(p(z)\) are the marginal probability functions of \(X\) and \(Z\) respectively.

From Equation 1, we can find that maximizing the mutual information between \(X\) and \(Z\) is equivalent to maximizing the Kullback-Leibler divergence between the joint distribution of \(p(x, z)\) and the product of marginal distribution of \(p(x)\) and \(p(z)\).

Following [10], the general form of f-divergence can be approximated by

\[
D(P||Q) = \sup_{T \in T} (\mathbb{E}_P[T(X)] - \log \mathbb{E}_Q[f^*(T(X))]),
\]

where \(P\) is the joint distribution \(p(x, z)\) and \(Q\) is the product of marginal distribution \(p(x)p(z)\). \(T\) is an arbitrary class of functions \(T : X \mapsto \mathbb{R}^\ast\) and \(f^*\) is the convex conjugate function of a generator function \(f\). Since \(D(P||Q)\) can be approximated by the supremum of the difference between two expectations, we can choose to maximize

\[
F(\omega) = \mathbb{E}_P[T_\omega(X)] - \log \mathbb{E}_Q[f^*(T_\omega(X))]
\]

instead of maximizing \(D(P||Q)\). \(T_\omega\) is a neural network parametrized by \(\omega\). More specifically, \(T_\omega\) can be represented in the form \(T_\omega(x) = g_f(V_\omega(x))\) where \(g_f\) is specific to the f-divergence used. Since Kullback-Leibler divergence is not upper bounded, we use Jensen-Shannon divergence as a surrogate to estimate the mutual information. Thus, we can replace \(f^*(t)\) with \(-\log(2 - e^t)\) and choose \(g_f(v) = \log(2) - \log(1 + e^{-v})\). Ignoring the constant term, we obtain

\[
F(\omega) \approx \mathbb{E}_P[\sigma(V_\omega(x))] - \log \mathbb{E}_Q[1 - \sigma(V_\omega(x))]
\]

where \(\sigma(v) = \frac{1}{1+e^{-v}}\) and \(\sigma(V_\omega(x))\) is represented by network \(D\) in Figure 2(b). From Equation 4, we can find that the maximization of \(F(\omega)\) will result in the network \(D\) outputting one for samples from the joint distribution and zero for samples from the product of the marginal distributions.

3.2. Constructing sample pairs

In Equation 4, we still need to estimate two expectations. In the first term the samples are sampled from the joint distribution and in the second term the samples are sampled from the product of marginal distributions. Since we are estimating the mutual information between adjacent convolutional layers \(H_i\) and \(H_{i+1}\), we firstly resize \(H_{i+1}\) to the size of \(H_i\). The reason we choose up-sampling instead of down-sampling is that we want to maximize the amount of information propagated to the next layer. Then sampling from the joint distribution \(p(h_i, h_{i+1})\) could be achieved by sampling feature vectors at the same spatial location on the convolutional feature maps. For sampling from the second distribution, we can firstly randomly sample a feature vector from \(H_i\) and then randomly sample another feature vector from \(H_{i+1}\). For each sample pair, the two feature vectors are concatenated as a single vector. The details are shown...
in figure 2. \( \sigma(V_{\omega}(x)) \) is represented by network D and the maximization of Equation 4 will optimize network D to distinguish the sample pairs from the two distributions.

### 3.3. Information flow maximization

When stacking convolutional layers, we are potentially losing information. According to DPI, we have the mutual information between the input X and the hidden layer \( H_l \) 
\[
I(X, H_l) > I(X, H_{l+1}) > \ldots > I(X, H_n).
\]
Suppose we are given a dataset for classification and the data representation can be decomposed into three disentangled features \( z_{id0}, z_{id1} \) and \( z_v \) where \( z_{id0} \) and \( z_{id1} \) can be used for classification and \( z_v \) describes some random variations that are shared across categories. Ideally, we can use these three features to perfectly reconstruct the input data. When we are training a model for the target classification task, the information about \( z_v \) will be gradually discarded from the information flow which is as expected. However, if the classification task is biased towards one of the id features, say \( z_{id0} \), we may unexpectedly lose the information of \( z_{id1} \) as well in the information flow. This is because during model training if \( \frac{\partial L}{\partial L_{id}} \) is much larger than \( \frac{\partial L}{\partial L_{id}} \), \( z_{id0} \) will get more strengthened than \( z_{id1} \). Finally, our model will rely only on \( z_{id0} \) for classification. This behavior undermines the generalization capability of our model especially when the test task depends on \( z_{id1} \) for classification.

To reduce the loss of information in the information flow, we propose to combine classification loss with the information flow maximization (IFM) loss, i.e.

\[
L = L_{clf} - \sum_{l=1}^{L} F_l(\omega),
\]

where \( L_{clf} \) is the classification loss (e.g. the softmax loss) and \( L \) is the number of layers that used to calculate the information flow.

Although some task-irrelevant information may also be involved in the final representation, the training process will let the discriminative information dominate the representation. Thus the classifier can make predictions based on more informative features.

### 4. Experiments

#### 4.1. Dataset

The dataset we used in evaluation is the shiftedMNIST dataset introduced in [9]. It is a modified version of MNIST dataset. Ten texture images are randomly selected from a texture dataset [3] and applied on the digit as its background. We spare one-fifth of the original MNIST training set to construct the validation set. In the training set, each digit is associated with a fixed type of texture. For example, for digit 1, its background patch is sampled from texture 1.

For digit 2, its background patch is sampled from texture 2, etc. However, in the validation set and test set, the digit is associated with a random texture. In other words, the texture id and the digit id are the same for given a training image while they are not necessarily the same for a given validation or test image. Some examples from the shiftedMNIST dataset are shown in Figure 3.

#### 4.2. Implementation details

The details of the classification network and the network D are shown in Table 1 and Table 2. The learning rate is 0.01. Mutual information is estimated for (conv1, conv2), (conv2, conv3) and (conv3, conv4). The size of the input image is \( 32 \times 32 \). Both the classification network and the network D are trained in an end-to-end way simultaneously.
4.3. Evaluation protocol

For the shiftedMNIST dataset, one may argue that only the digit feature should be considered as the correct feature for label prediction in the training set. However, as stated in [7], the digit feature can be viewed as a kind of human prior. For our model, it does not have such a prior so that both the digit feature and the texture feature may be viewed as discriminative features. It leaves to the optimization dynamics to choose which feature as the final predictor. Note that the digit label and texture label are identical for a given training image. In the training stage, we select one model with best validation accuracy for digit classification and another model with best validation accuracy for texture classification to observe how the optimization dynamics influences the knowledge learning process. Note that the optimal test classification accuracy should be around 55% since the classification model is not aware of whether the test task is a digit classification task or a texture classification task. So we expect it to learn both features equivalently.

4.4. Results

The classification results are shown in Table 3. In this section, we train a baseline model without IFM blocks. Model $Baseline_{digit}$ achieves best digit validation accuracy and the model $Baseline_{texture}$ achieves best texture validation accuracy. The test accuracies are shown in the first two rows in Table 3. For both models, the prediction accuracies on the digit are slightly above 10% which is quite similar to random guess. It means that both models ignore the digit structure as the discriminative feature. The prediction accuracies on texture are above 95% which means the final representations are dominated by the texture features. The results of the baseline models demonstrate that if the model is trained in the vanilla way it only learns partial discriminative features and ignores other correlated features. In this experiments, the baseline models are sensitive to texture features which is in accordance with the observations in [5].

The benefit of applying IFM is shown in the bottom two rows. It can be found that the classification accuracies of digit are much higher than that of the baseline model. It indicates that our models indeed learn the digit structure as the discriminative feature. The reason for the test digit accuracy of $Ours_{digit}$ being much lower than the texture accuracy is that digit structure features are more difficult to learn than texture features. When the texture features are well learned (with high texture validation accuracy), the learning of digit features may only reach its halfway. Then IFM loss prevents texture features from dominating the final representation and provides more structure information. Thus test texture accuracy decreases and digit accuracy increases. $Ours_{digit}$ also outperforms the model in [9] which is a flow based model with no information loss. It implies that our IFM can be potentially viewed as an alternative way to flow-based models to reduce information loss in deep networks.

5. Conclusion

In this work, we propose to maximize the information flow in convolutional neural networks as a kind of regularization term. The benefit of this regularization is that we can find correlated features that are difficult to be disentangled. Thus, the learned representations are more informative and generalizable than that learned in conventional training without this information regularization term. Our future work will focus on how to apply the proposed information flow maximization on natural image classification tasks.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Network details</th>
</tr>
</thead>
<tbody>
<tr>
<td>conv1</td>
<td>Conv(32,3,3)-BN-leakyReLU</td>
</tr>
<tr>
<td></td>
<td>Maxpool(2,2)</td>
</tr>
<tr>
<td>conv2</td>
<td>Conv(64,3,3)-BN-leakyReLU</td>
</tr>
<tr>
<td></td>
<td>Maxpool(2,2)</td>
</tr>
<tr>
<td>conv3</td>
<td>Conv(128,3,3)-BN-leakyReLU</td>
</tr>
<tr>
<td></td>
<td>Maxpool(2,2)</td>
</tr>
<tr>
<td>conv4</td>
<td>Conv(128,3,3)-BN-leakyReLU</td>
</tr>
<tr>
<td></td>
<td>Maxpool(2,2)</td>
</tr>
<tr>
<td>f</td>
<td>FC(128×2×2, 10)</td>
</tr>
</tbody>
</table>

Table 1. Details of the network D. $N$ indicates the input dimension of the concatenated feature vector.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Network details</th>
</tr>
</thead>
<tbody>
<tr>
<td>fc1</td>
<td>FC(N, 256)-BN-leakyReLU</td>
</tr>
<tr>
<td>fc2</td>
<td>FC(256, 128)-BN-leakyReLU</td>
</tr>
<tr>
<td>fc3</td>
<td>FC(128, 64)-BN-leakyReLU</td>
</tr>
<tr>
<td>fc4</td>
<td>FC(64, 1)-sigmoid</td>
</tr>
</tbody>
</table>

Table 2. Details of the network D. $N$ indicates the input dimension of the concatenated feature vector.

<table>
<thead>
<tr>
<th>Model</th>
<th>acc (digit)</th>
<th>acc (texture)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline_{digit}</td>
<td>12.44%</td>
<td>95.07%</td>
</tr>
<tr>
<td>Baseline_{texture}</td>
<td>12.05%</td>
<td>96.44%</td>
</tr>
<tr>
<td>iCE fi-RevNet [9]</td>
<td>40.01%</td>
<td>-</td>
</tr>
<tr>
<td>Ours_{digit}</td>
<td>54.54%</td>
<td>40.11%</td>
</tr>
<tr>
<td>Ours_{texture}</td>
<td>31.78%</td>
<td>69.00%</td>
</tr>
</tbody>
</table>

Table 3. Test classification accuracy from the baseline model, iCE fi-RevNet and ours.
References


