Project AI: Investigating the Learning Dynamics of Convolutional Neural Networks using the Mutual Information Neural Estimator

Nikita Tokovenko (12185892) **Lars Rigter** (12237019)[1]

Abstract

Recently, a new approach called Mutual Information Neural Estimator (MINE) is proposed to approximate the mutual information (MI) for high-dimensional data. In this work we have studied the consistency of MINE for estimating MI during the training process of Convolutional Neural Networks (CNNs). In particular, we are using MINE to measure the MI of each layer in a CNN with respect to its input and ground truth labels while training LeNet-5 on MNIST dataset. We show that the estimation of MI is highly dependent on the architecture and optimization of MINE.

1. Method

1.1. Mutual Information

The mutual information between two random variables X and Z can be seen as the gain of information on X you get once given Z.

$$I(X, Z) = H(X) - H(X|Z), \tag{1}$$

where $H(X)$ is the Shannon entropy of X and $H(X|Z)$ is the conditional entropy of X given Z. On the other hand, it can be seen as the cost of treating random variables being independent when they are not. Thus, mutual information can be calculated as Kullback-Leibler (KL-) divergence between the joint distribution, $\mathbb{P}_{XZ}$, and the product of the marginals $\mathbb{P}_X \otimes \mathbb{P}_Z$:

$$I(X, Z) = D_{KL}(\mathbb{P}_{X,Z} || \mathbb{P}_X \otimes \mathbb{P}_Z) \tag{2}$$

MI captures non-linear dependencies and can be seen as a measure for dependence (Kinney & Atwal, 2014).

[1]Supervisor: Maximilian Ilse, AMLab.
Nikita Tokovenko <nikita.tokovenko@student.uva.nl>
Lars Rigter <larsrigter60@gmail.com>.

1.2. The Mutual Information Neural Estimator

Theorem 1 *(Donsker-Varadhan representation). The KL divergence admits the following dual representation:*

$$D_{KL}(\mathbb{P}||\mathbb{Q}) = \sup_{T:\Omega \mapsto \mathbb{R}} \mathbb{E}_{\mathbb{P}}[T] - \log(\mathbb{E}_{\mathbb{Q}}[e^T]) \tag{3}$$

where the supremum is taken over all functions T such that the two expectations are finite.

By combining (2) with (3) MINE is defined as $T_\theta : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ parametrized by a deep neural network with parameters $\theta \in \Theta$ in order to find the lower-bounding statistical network (Mohamed Ishmael Belghazi & Courville, 2018), such that:

$$I(X, Z) \geq I_\theta(X, Z) \tag{4}$$

where I_θ is the neural information measure defined as

$$I_\theta(X, Z) = \sup_{\theta \in \Theta} \mathbb{E}_{\mathbb{P}_{XZ}}[T_\theta] - \log(\mathbb{E}_{\mathbb{P}_X \otimes \mathbb{P}_Z}[e^{T_\theta}]). \tag{5}$$

MINE, in turn, aims to maximize this lower-bound by estimating those expectations using empirical samples from $\mathbb{P}_{XZ}$ shuffled along the batch axis to get samples from the factorized marginals.

Details on the implementation of MINE are provided in Algorithm 1.

Algorithm 1 MINE

```
  $\theta \leftarrow$ initialize network parameters
  repeat
      Draw $b$ minibatch samples from the joint distribution: $(\boldsymbol{x}^{(1)}, \boldsymbol{z}^{(1)}), \ldots, (\boldsymbol{x}^{(b)}, \boldsymbol{z}^{(b)}) \sim \mathbb{P}_{XZ}$
      Draw $n$ samples from the $Z$ marginal distribution: $\bar{\boldsymbol{z}}^{(1)}, \ldots, \bar{\boldsymbol{z}}^{(b)} \sim \mathbb{P}_Z$
      Evaluate the lower-bound: $\nu(\theta) \leftarrow$
      $\leftarrow \frac{1}{b}\sum_{i=1}^{b} T_\theta(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)}) - \log(\frac{1}{b}\sum_{i=1}^{b} e^{T_\theta(\boldsymbol{x}^{(i)}, \bar{\boldsymbol{z}}^{(i)})})$
      Evaluate gradients:
      $\widehat{G}(\theta) \leftarrow \nabla_\theta \nu(\theta)$
      Update the statistics network parameters:
      $\theta \leftarrow \theta + \widehat{G}(\theta)$
  until convergence
```

1.3. The Information Plane

The information plane consists of MI values between the input and the layer output ($I(X, L)$) on one axis and MI between the target labels and the layer output ($I(Y, L)$) on the other axis. The information path in information plane satisfies (Shwartz-Ziv & Tishby, 2017):

$$I(X,Y) \geq I(L_1,Y) \geq \ldots \geq I(L_k,Y) \geq I(\widehat{Y},Y) \quad (6)$$

$$H(X) \geq I(X,L_1) \geq \ldots \geq I(X,L_k) \geq I(X,\widehat{Y}) \quad (7)$$

1.4. MINE architecture

Difficulty of applying MINE to CNNs arises from the different dimensionality of all representations involved: X, Y and L. There are several ways to achieve a mapping from the joint space to a single scalar. Therefore, different architectures of MINE were studied:

- Architecture 1 : Layer 1: flatten input/target and layer output and concatenate along feature dimension. Layer 2: Linear module of size 10 with ReLU activation. Layer 3: Dense of size 1.
- Architecture 2: Layer 1: apply convolution to 2D input/layer output and use ReLU activation for 1D input/layer output. Layer 2: Linear module of size 10 with ReLU activation. Layer 3: Dense of size 1.
- Architecture 3: Layer 1: flatten input/target and layer output, apply non-linearity and concatenate along feature dimension. Layer 2: Linear module of size 10 with ReLU activation.Layer 3: Dense of size 1.
- (addition) Add noise to the layer output.
- (addition) Different number of hidden units of the Linear module.

1.5. Training procedure

The convolutional network architecture used for the purposes of this research project was LeNet-5. Exact architecture can be found in Table 1.5. For our experiments we estimated the value of the mutual information after passing the first pooling layer (maxpool1), the second pooling layer (maxpool2), the first linear layer followed by the ReLU activation (relu3), the second linear layer followed by the SoftMax function (sm1).

We trained LeNet-5 on MNIST dataset to classify handwritten digits on the input image. We set the learning rate to 0.0003. For every 4^{th} epoch of training LeNet-5, the MI is estimated by training MINE either from scratch (re-initialization) or by fine-tuning the model obtained from the previous run. In total LeNet-5 was trained for 21 epochs, MINE was trained 6 times.

layer	shape
convolution1 + relu	$1 \times 20 \times 5$
maxpool1	2×2
convolution2 + relu	$20 \times 50 \times 5$
maxpool2	2×2
fc1 + relu (relu3)	500
fc2 + softmax (sm1)	10

Table 1. Architecture of LeNet-5

The mutual information was approximated pairwise both for input (X) and target (Y) with outputs of maxpool1, maxpool2, relu3 and sm1.

2. Results

2.1. Different architectures of MINE

MI between input and layer output		
MINE architecture	addition	Result
Architecture 1	-	Exploding gradients
Architecture 1(2)	Noise added to layer output	Converged approximation of the lower bound
Architecture 1(3)	20 hidden units for linear module and noise added to layer output	Converged approximation of the lower bound
Architecture 2	-	Underestimated MI
Architecture 3	-	Exploding gradients
Architecture3(2)	Noise added to layer output	Underestimated MI
MI between Target and layer output		
Architecture 1	-	Exploding gradients
Architecture 1(3)	Dense+ReLU of size 100 and noise added to layer output	Converged approximation of the lower bound

Table 2. Different architectures of MINE

Based on table 2.1 it can be noticed that the architecture of MINE has an effect on its consistency. When the first MINE layer contains convolutions, lower bound of MI is underestimated. When the first layer contains a non-linearity the approximator becomes much stronger, and without adding noise gradients can explode. Also adding more non-linearities after the first MINE layer causes gradient explosion. This can be stabilized by adding noise to the CNN's hidden layer output (L), thus constraining the value of true MI. In the end, the lower bound for estimation is stable when adding noise to the layer output and flattening X/Y and L, followed by a ReLU module of size 20. MINE for target and layers output seems to be strong enough and stable when flattening the input, adding noise to L and adding a ReLU activation. For the experiments, the architecture flatten-Dense+ReLU-Dense for a different amount of hidden units was used.

2.2. Accuracy of LeNet-5

Epoch	Accuracy
0	70%
4	83%
8	91%
12	94%
16	96%
20	97%

Table 3. Accuracy on the test set per epoch of LeNet-5

As can be seen in table 2.2 the performance of the target CNN gradually increases over epochs. Such a gradual increase is caused by the usage of a decreased learning rate (0.0003).

2.3. Transferring weights of MINE

Transferring weights of MINE between different runs can serve the purpose of acceleration the learning process of the estimator. The process of training MINE to estimate $I(Y, \text{relu3})$ with transferring weights is visualized on Figure 2. It can be noticed that the lowerbound for MI converges almost after the first epoch for run 3 to 6. In comparison, learning curves for re-initializing MINE every run can be seen on Figure 1. It can be noticed that it takes around 30 epochs for MINE to converge when MINE is re-initialized every run.

2.4. Mutual information during the training of LeNet-5

The information plane was reconstructed with two different seeds as can be seen on Figures 3 and 4. It can be noticed that estimates for $I(., \text{relu3})$ and $I(., \text{sm1})$ show consistent behavior for different seeds ($I(X, \text{relu3} \geq I(X, \text{sm1})$,

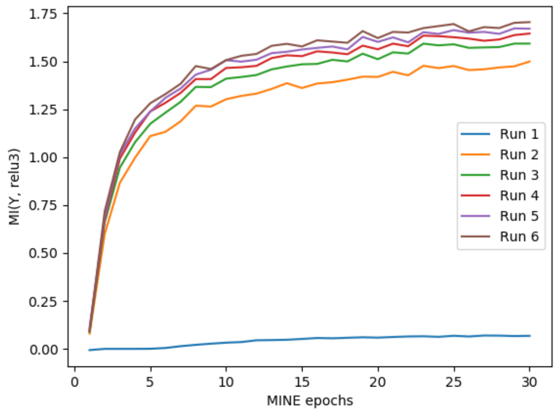

Figure 1. Training LeNet-5 and training MINE (**without** transferring weights+20 hidden cells) to estimate MI between target and ReLU3 where 1 run is made 4 epochs of training LeNet-5

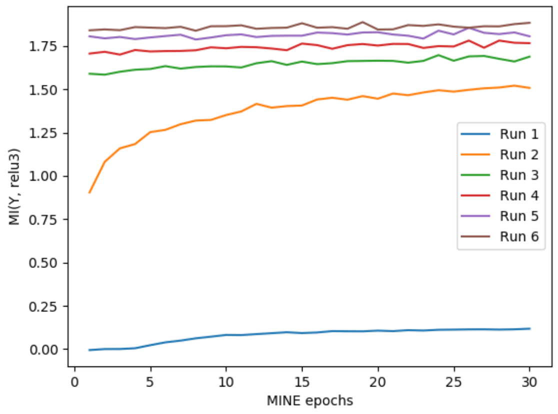

Figure 2. Training LeNet-5 and training MINE (**with** transferring weights+20 hidden cells) to estimate MI between target and ReLU3 where 1 run is made 4 epochs of training LeNet-5

$I(Y, \text{relu3}) \leq I(Y, \text{sm1})$). This makes sense since we assumed that Y is more dependent on output of sm1 than on output of relu3. Estimations of $I(Y, \text{maxpool1})$ and $I(Y, \text{maxpool2})$ seem also to be consistent. Based on Figures 3 and 4 it can be noticed that estimations of $I(X, \text{maxpool1})$ and $I(X, \text{maxpool2})$ differ quite a lot. Besides, $I(X, \text{maxpool1}) \leq I(X, \text{maxpool2})$. This is counter-intuitive since we expected that output of maxpool1 is more dependent on X than the output of maxpool2 is. Finally, on Figure 5 it can be seen that using more units for the hidden layer yields intuitive correct MI flows for all the layers. However, during training, the estimation of lower bound did not converge for all runs and gradients also exploded in some cases. MI estimation is almost equal for maxpool1 and maxpool2 whereas one could observe that $I(X, \text{maxpool2})$ is higher as shown on Figure 4.

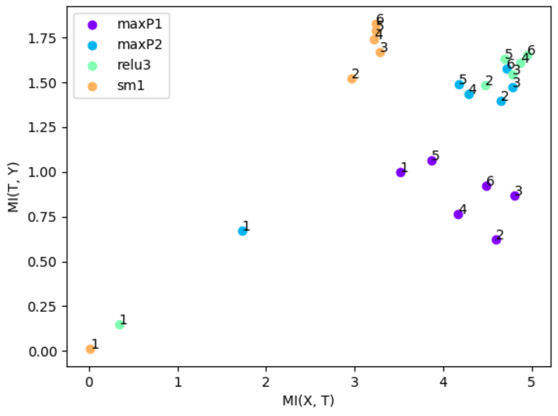

Figure 3. Run1: Training LeNet-5 and estimating $I(X, L)$ and $I(Y, L)$ with MINE (no weight transfer and 20 hidden cells) for every 4 epochs. Different numbers next to dots correspond to different MINE learning runs over stages of training LeNet-5

3. Conclusion

We have experienced that adding learnable layers to MINE architecture in order to handle input with different dimensionalities causes instabilities and/or underestimation. This holds for the case of estimating MI in LeNet-5. Besides, adding learnable layers to MINE after flattening the input does increase the power of MINE. However, MINE can overestimate, meaning the lower bound estimation process does not converge and at the end gradients explode. Moreover, fine-tuning and noise addition is very important for the consistency of MINE estimator. Even with fine-tuned hyperparameters, MINE can be unstable, especially for measuring mutual information between target CNN's input and output of convolutional layers followed by pooling.

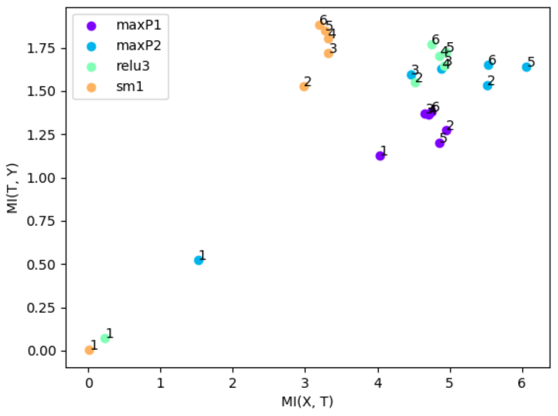

Figure 4. Run2: Training LeNet-5 and estimating $I(X, L)$ and $I(Y, L)$ with MINE (no weight transfer and 20 hidden cells) for every 4 epochs. Different numbers next to dots correspond to different MINE learning runs over stages of training LeNet-5

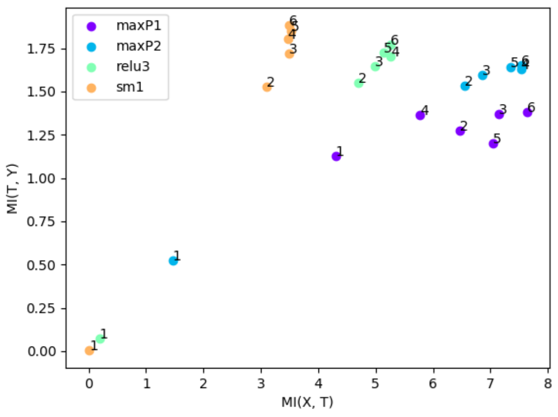

Figure 5. Run3: Training LeNet-5 and estimating $I(X, L)$ and $I(Y, L)$ with MINE (no weight transfer and 100 hidden cells) for every 4 epochs. Different numbers next to dots correspond to different MINE learning runs over stages of training LeNet-5

In future research, our findings can be used to properly set the architecture of MINE and study the consistency of MINE to estimate the mutual information for different network architectures and for different data sets. Besides, it can be studied how MINE can be made to be more stable for capturing information flow while training CNNs.

References

Kinney, J. B. and Atwal, G. S. Equitability, mutual information, and the maximal information coefficient. *Pro-*

ceedings of the National Academy of Sciences, 111(9): 3354–3359, 2014.

Mohamed Ishmael Belghazi, Aristide Baratin, S. R. S. O. Y. B. D. H. and Courville, A. Mutual information neural estimation. *International Conference on Machine Learning (ICML)*, pp. 530539, 2018.

Shwartz-Ziv, R. and Tishby, N. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.