Multiple Scattering Media Imaging via End-to-End Neural Network

Ziyang Yuan∗ Hongxia Wang†

Abstract

Recovering the image of an object from its phaseless speckle pattern is difficult. Let alone the transmission matrix is unknown in multiple scattering media imaging. Double phase retrieval is a recently proposed efficient method which recovers the unknown object from its phaseless measurements by two steps with phase retrieval.

In this paper, we combine the two steps in double phase retrieval and construct an end-to-end neural network called TCNN(Transforming Convolutional Neural Network) which directly learns the relationship between the phaseless measurements and the object. TCNN contains a special layer called transform layer which aims to be a bridge between different transform domains. Tested by the empirical data provided in[1], images can be recovered by TCNN with comparable quality compared with state-of-the-art methods. Not only the transmission matrix needn’t to be calculated but also the time to recover the object can be hugely reduced once the parameters of TCNN are stable.

Keywords: neural network, phase retrieval, multiple scattering media, end-to-end

1 Introduction

The light incident on the multiple scattering media will suffer from multiple reflection. Thus when observing the object through the multiple scattering media with coherent light, the speckle pattern displayed on the far side of the scatter isn’t similar with the object which can be seen from Figure 1. Because the wavefront interferences with itself destructively when passing through the multiple scattering media, many details about the object get lost. At the same time, the transmission matrix about this complex media is hard to be analyzed and constructed. Moreover, the detector such as CMOS or CCD can only record the intensity of the speckle pattern. As is well known, recovering the object from its phaseless measurements called phase retrieval is an ill-posed inverse problem. Overall, it’s hard to recover image of the object from multiple scattering media. Considering the importance of this problem in applications, there are a batch of techniques to deal with it such as TOF(Time of Flight) method[2], multi-slice light-propagation method[3], strong memory effect method[4], holographic interferometry method[5], temporally modulated phase method[6] and double phase retrieval method[7][8]. Interested readers can refer to [1] for a review.

Comparing to other methods, the double phase retrieval method can alleviate from the depth
and complexities of the scatterer besides this method is able to reconstruct image after capturing only a single speckle pattern once the transmission matrix is estimated. The mathematical formulation of double phase retrieval is:

\[
\text{Find } \mathbf{A}, \mathbf{x} \\
\text{s.t. } |\mathbf{A}^*\mathbf{x}|^2 = \mathbf{b},
\]

where \( \mathbf{A} \in \mathbb{C}^{n \times m} \) is the transmission matrix, \( \mathbf{x} \in \mathbb{C}^n \) is the signal of interest. \( \mathbf{b} \) is the measurement, \( \ast \) is the conjugate and transpose. \( |\cdot| \) is the element-wise absolute. The double phase retrieval method has two main steps: estimating \( \mathbf{A} \) firstly, then recovering \( \mathbf{x} \) based on \( \mathbf{A} \) from the first step. As its name suggested, the core of double phase retrieval is to solve several phase retrieval problems.

Phase retrieval, namely recovering the object from the phaseless measurements which arises from lots of applications such as holographic imaging, coherent diffraction imaging and astronomy, etc. The mathematical formulation of phase retrieval is same with formula (1.1), but the transmission matrix \( \mathbf{A} \) is known in advance. The uniqueness of the problem is often up to a global phase factor. During several decades, lots of theoretical analyses and numerical algorithms have come up to solve the phase retrieval problem. Gerchberg and Saxton method \[9\] and Fienup method \[10\] are the two types of classical alternating methods to find the solutions of phase retrieval problem. They have been utilized in various of applications. In \[11\], it came up with a gradient descent based method called Wirtinger flow to search for the solutions of the non-convex problem. Theoretical analyses also guarantee the convergence to the global optimum when each column of \( \mathbf{A} \) satisfies \( \mathbf{a}_i \sim_{i.i.d} \mathcal{N}(0, \mathbf{I}), i = 1, \cdots, m \). Based on this, several works were came up to decrease the sampling complexity besides increasing the recovery probability. Convex relaxation can also be utilized to relief the phase retrieval problem. Phaselift and Phasemax are the two different representatives of convex methods for dimension lifting and constraints relaxation \[12\] \[13\] \[14\]. In theoretical analyses, when \( m \geq 2n - 1 \) or \( m \geq 4n - 4 \), uniqueness can be guaranteed for phase retrieval when \( \mathbf{A} \) is real or generic complex \[15\] \[16\]. At the same time, oversampling can relief the illness of phase retrieval problem and improve the performance of corresponding algorithms by numerous numerical tests.
The double phase retrieval can also be called blind phase retrieval. In this case, the transmission matrix $A$ and $x$ is unknown. The condition of it is worse than phase retrieval. As a result, in the first step, $A$ will be evaluated from $k$ known images $x^{(i)}, i = 1, \cdots, k$ and their corresponding known intensity measurements $b^{(i)}, i = 1, \cdots, k$. In this circumstance, the model can be defined as below:

$$\text{Find } A \text{ s.t. } |A^*x^{(i)}|^2 = b^{(i)}, i = 1, \cdots, k.$$  \hspace{1cm} (1.2)

As a result, define $X = [x^{(1)}, x^{(2)}, \cdots, x^{(k)}]$, $B = [b^{(1)}, b^{(2)}, \cdots, b^{(k)}]$, each column of $A$ namely $a_j$ can be estimated accurately through a series of phase retrieval problems.

$$\text{Find } a_j \text{ s.t. } |X^*a_j|^2 = B^*_j, j = 1, \cdots, m,$$

where $B^*_j$ is the $j$th column of $B^*$. After obtaining the evaluation of $A$, utilizing standard phase retrieval methods mentioned above, the unknown $x$ can be recovered from its measurements $b$. In [1], they built up different experimental setups to apply multiple-scatter media imaging, then utilizing the double phase retrieval method based on different phase retrieval algorithms to recover the image. Numerous tests show the good performance and robustness of this method.

Though the double phase retrieval method is efficient, it costs lots of computational resources to solve series of phase retrieval problem. It must evaluate the transmission matrix $A$ firstly which demands plenty of training data and the large computational burdens for solving phase retrieval problem. In the second step, solving the classical phase retrieval method also needs too much computational cost. Besides it seems a little complicated to solve the problem in two steps. Thus time saved, explicit and efficient method need to be devised.

Deep learning has reached much attention since Alexnet won the champion in the ISVRC 2012. Since then, lots of layer structures and optimization methods camp up to accelerate the development of deep neural network. With the aid of the hardwares such as GPU and series of famous open access projections such as Tensorflow, Pytorch and Caffe, deep learning have been successfully applied in Object detection, autonomous vehicles, Signal processing such as Voice detection and Voice synthetic, inverse problem such as MRI[17], holography[18] and super-resolution[19]. For phase retrieval, there are some previous works which also utilize the neural network to increase the imaging quality. In [20], they built a neural network to diminish the effect of the twin image. Compared to the classical methods, this neural network only requires the measurements obtained in a single distance besides having a competitive performance. In [21], it also came up with a neural work to increase the resolution of the image in lensless coherent diffraction image. This neural network is also an end-to-end neural network which can directly transform the diffraction pattern into the image.

Compared to those works above, the transmission media in our test is worse besides the speckle pattern bear no resemblance to the image. Those factors make it more difficult to recover the object for neural network. In this paper, we combine two steps in double phase retrieval together by deep neural network called TCNN(Transformation Convolution Neural Network) to directly learn the relationship between the intensity of the speckle pattern and the image of object. Applying TCNN into multiple-scatter media imaging, what we need do is to train the neural network which fully utilizes the training sets of double phase retrieval method in the first step. Once the training of
network is finished, the recovered image can be obtained immediately by inputting the intensity of speckle pattern into neural network without calculating the transmission matrix. Tests using the multiple-scattering-media imaging data in [1] demonstrate that TCNN can have a competitive performance with the state-of-the-art. When the training procedure of TCNN has been completed, the time required by TCNN is much less than state-of-art for solving the phase retrieval problem when \( A \) is estimated. Besides, TCNN can be refined based on the well-trained network if more training data is available. But double phase retrieval method has to be calculated from all the training data again so that the transmission matrix can be updated. TCNN is a special network which is devised to solve the inverse problem. Thus special structure called transforming layer is constructed in TCNN which can help the neural network learn the relationship between the transforming domain and object domain so that TCNN can recover the image efficiently.

The reminders of this paper are organized as below. In section 2, the details of the experiment setup and TCNN are given. In section 3, the results of the experiment for TCNN are given. Section 4 is the conclusion.

2 The feasible of neural network for solving double phase retrieval

Before we describe TCNN, the feasible analyses of neural network to solve double phase retrieval is built. Neural network wants to approximate function \( g \) so that \( x \) can be estimated directly from \( b \).

\[
g_\Theta : b \rightarrow x, \tag{2.1}
\]

where \( \Theta \) is the set of parameters in the neural network which are learned via enough known \( b \) and \( x \). Initially, the existence of the operator \( g \) must be discussed.

The operator \( f : x \rightarrow b \) must be injective or \( g \) doesn’t exist, where \((f(x))(i) = |a_j^*x|^2\). We can clearly see that when \( x \in \mathbb{R}^n \), \( f(x) = f(-x) \) besides if \( x \in \mathbb{C}^n \), \( f(x) = f(cx), |c| = 1 \). As a result, the injective of \( f \) can be satisfied only if \( x \) defined up to a global phase factor. Thus we consider the map \( f : \mathbb{R}^n/\{\pm 1\} \rightarrow \mathbb{R}^m \) when \( x \) is real, \( f : \mathbb{C}^n/T \rightarrow \mathbb{R}^m \) when \( x \) is complex(where \( T \) is the complex unit circle). Next we introduce the theorem below to guarantee the injective of \( f \).

**Theorem 2.1.** [15] Assuming there is no noise, \( a_j, j = 1, \cdots, m \) are generic frame vectors, when \( a_j \in \mathbb{R}^n \), \( x \in \mathbb{R}^n \) or \( a_j \in \mathbb{C}^n \), \( x \in \mathbb{C}^n \), if \( m \geq 2n-1 \) or \( m \geq 4n-2 \), \( f \) is injective.

Because the light passing through the multiple-scattering media which may be frost glass or painted wall. In classical analyses, the rows of transmission matrix \( A \) of those media are often assumed satisfying the gaussian or sub-gaussian distribution which are generic vectors. So the property of transmission matrix in Theorem 2.1 can be satisfied. Before training the network, it must sift the dataset so that every \( b^{(i)}, i = 1, \cdots, k \) is different to guarantee injective.

Now that the existence of the inverse function \( g \) is guaranteed. As a result, we can build up the universality theorem below:

**Theorem 2.2.** If the inverse function \( g \) of phase retrieval \( f(f : \mathbb{R}^n/\{\pm 1\} \rightarrow \mathbb{R}^m \) or \( f : \mathbb{C}^n/T \rightarrow \mathbb{R}^m \)) exists, then \( g \) can be approximated by \( g_\Theta \) with any desired degree of accuracy.

Proof. When \( x \in \mathbb{R}^n \), \( f : x \rightarrow b \) is a continuous and borel measurable function. As a result, if \( g \) exists, \( g \) is also a borel measurable function. Utilizing the universality theorem in [22], \( g \) can be
approximated by $g_{\Theta}$ with squashing functions to any desired degree of accuracy as if hidden units are sufficient.

When $x \in \mathbb{C}^n$, we split $x$ into two parts $x_1$ and $x_2$ uniquely which satisfy $x_1 + jx_2 = x$ where $j = \sqrt{-1}$. Because $g$ exists, as a result, let $g^{(1)} = (g + \overline{g})/2$, $g^{(2)} = (\overline{g} - g)j/2$, so $g^{(1)}(b) = x_1$, $g^{(2)}(b) = x_2$. Because $g^{(1)}$ and $g^{(2)}$ are all borel measurable function. Thus, utilizing the results when $x \in \mathbb{R}^n$, $g^{(1)}$ and $g^{(2)}$ can be approximated by $g^{(1)}_\Theta$ and $g^{(2)}_\Theta$. So $x \approx g^{(1)}_\Theta(b) + jg^{(2)}_\Theta(b)$. □

Above all, utilizing the universality of neural network, when the training set is in $\mathbb{R}^n/\{\pm 1\}$ or $\mathbb{C}^n/\{T\}$, it is possible to train a neural network to approximate this inverse function $g$. In this paper, we build a neural network called TCNN to train $g_{\Theta}$ from dataset. The core of TCNN is to utilize two sections $g_{\Theta_1}$ and $g_{\Theta_2}$ to approximate the inverse function $g$ which can simulate the imaging procedure in multiple scattering media, namely,

$$g(b) \approx g_{\Theta}(b) = g_{\Theta_2}(g_{\Theta_1}(b)). \quad (2.2)$$

For $g_{\Theta_1}$, features of intensity of speckle patterns $b$ are learned from different scales which can be seen from Figure 2. We can see that the details from different scales of speckle pattern are learned (the features in Figure 2 are resized to keep the same). Then deconvolution procedures decode those features into a new element $g_{\Theta_1}(b)$ in the transform domain so that it can be approximated to $x$ by linear transform $g_{\Theta_2}$. The diagram of TCNN can be viewed in Appendix.

The multiple flow structure is utilized in TCNN which can learn the features of the input in different scales. The input of neural network is decimated by the downsampling Block which is constructed by one convolutional layer with batch normalization, Relu activation function and one
maxpooling layer. The maxpooling layer here decreases the dimension of the input. Specifically the input is downsampled by $\times 2, \times 4, \times 8$ respectively. Then, the four different tensors will be successively passed through five residue blocks, each block contains two convolutional layers with batch normalization and a shortcut between the input and output of the block. The shortcut can accelerate the convergence of TCNN. After the five residue blocks, the high order features in different scales are learned. Then, we will decode those features to generate four different tensors keeping the same size with the input. So there will be 3, 2, 1 upsampling blocks for each tensors respectively. In each upsampling block, there will be one convolutional layer with batch normalization and one deconvolutional layer. The deconvolution in this paper adopts the way of upsampling in [20] for super-resolution. Because this method can alleviate the effect of zero padding by traditional deconvolution besides fully utilizing the information in the network. After those upsampling blocks, the fused tensor is obtained which will pass through the transformation block instituted by one convolutional layer and transformation layer. The transformation layer is in fact a full connection layer which acts as a linear transformation between the transform domain and object domain. In the test, to alleviate the influence of over-fit, the units in transformation layer are randomly neglected. The details of TCNN can be found in Appendix.

3 The test of empirical data

The experiment setup displayed in this paper is made by Rice University in [1]. Here, we only display the experiment setup for phase only SLM. As shown in Figure 3, a spatially filtered and collimated laser beam($\lambda = 632.8nm$) illuminates an SLM from Holoeye. It is a reflective type display(LC2012) with 1024 $\times$ 768 resolution and 36 micrometer size square pixels. It can modulates the phase of the beam before the lens $L(f = 150mm)$, this lens can focus the beam onto the scattering medium which is a holographic 5 degree diffuser from Thorlabs. Then a microscope objective(Newport, X10, NA:0.25) is used to image the SLM calibration pattern onto the sensor which is the Point Grey Grasshopper 2 with pixel size 6.45 micrometer. Because the phase only SLM is 8 bit, it modulates the wavefront by an element of \{\begin{align*} e^{0}, e^{2\pi j \frac{1}{256}}, \ldots, e^{2\pi j \frac{255}{256}} \end{align*}\}. In the test, the phase modulation is restricted to \{0, $\pi$\}, which means the value of the $x$ is \{−1, 1\}. For the amplitude only SLM, set the source pixel as either completely off, 0 , or completely on ,1. then the value of $x$ is \{0, 1\}. For the details of the experimental setup, readers can refer to [1].

The test data can be download from [https://rice.app.box.com/v/TransmissionMatrices](https://rice.app.box.com/v/TransmissionMatrices). Two types of the data are utilized in the experiment. For amplitude only SLM, the size of the image is $16 \times 16$ or $64 \times 64$. For the phase only SLM, the size of the image is $40 \times 40$. Experiments are applied on desktop computer with GPU NVIDIA 1080 and the CUDA edition is 9.0. Some hyper parameters of the test can be found in Table 1. We imply TCNN using deep learning framework tensorflow. The loss function is chosen as the mean squared error(MSE) between the outputs of TCNN and corresponding benchmarks in the training set. The weights of TCNN are trained by the back propagation using ADAM. The initial learning rate of ADAM is $10^{-3}$ which decays with the factor of 0.85 after each epoch. The total number of epoch of the training procedure is 100. To preprocess the data in this experiment, we firstly sift the data so that every image and corresponding speckle pattern is unique. Then we trained TCNN for three different sizes of data.

We compare TCNN with state-of-the-arts in literature such as GS[9], WF[11], PhaseLift[12], Prvamp[1]. TCNN can directly recover images from those speckle patterns without estimating the transmission matrix. For other methods, we utilize the transmission matrix $A$ obtained by Prvamp
for phase retrieval. The results are shown in Figure 4 and Figure 5.

Table 1: Some hyper parameters in TCNN

|                        | 16 × 16 | 40 × 40 | 64 × 64 |
|------------------------|---------|---------|---------|
| The number of the training sets | 3050    | 3050    | 35000   |
| The number of the validation sets | 22      | 20      | 200     |
| The number of test sets    | 5       | 5       | 6       |

From Figure 4 and Figure 5 we can observe that pictures recovered by TCNN are competitive with state-of-arts. It can recover the outline of the object although some details are lost just like other methods. In fact, it is very hard to recover those images with high quality from speckle patterns. Because multiple scattering media deteriorates the imaging procedure besides the phase information lost by CCD is important to recover image. Moreover, the noise and system error also exist in the experiment. But Table 1 shows the time cost by TCNN is far less than other methods. It can save the time at least ten folds even 100 folds when 64 × 64 which demonstrates the advantages of TCNN to realize real time imaging. Moreover TCNN only performs an end-to-end recovery but other methods must estimate the transformation matrix in advance then getting the evaluation by phase retrieval. Thus, experiments above fully illustrates the power of TCNN to recover the image from the intensity of speckle pattern.

During the training step, the training error and validation error of amplitude only SLM 16 × 16 are given in Figure 6. Considering the large quantities of training set, training error is the MSE of one of the input in training set by random so there is some oscillations for training error. The validation error is the mean MSE for the validation set. From Figure 6, we can find the solution quickly converges to the local optimum, after 20 epochs, there is little fluctuation for both errors. Besides the validation error is very close to the training error which fully demonstrates the influence
Figure 4: The reconstruction of various methods for amplitude only SLM with image size $16 \times 16$
Figure 5: The reconstruction of various methods for phase only SLM with image size $40 \times 40$
Figure 6: The training error and validation error of over-fit got controlled.

Table 2: Time cost per image by different methods

| Method    | $16 \times 16$ | $40 \times 40$ | $64 \times 64$ |
|-----------|----------------|----------------|----------------|
|           | Time(s) iteration | Time(s) iteration | Time(s) iteration |
| WF        | 4.11 100          | 19.49 100       | 289.03 100     |
| GS        | 3.86 100          | 18.86 100       | 43.48 100      |
| PhaseLift | 239.50 100        | — 500           | — 1000         |
| PRVAMP    | 3.41 100          | 17.40 100       | 44.52 100      |
| TCNN      | 0.103 —           | 0.117 —         | 0.121 —        |

4 Conclusion

In this paper, we build a deep neural network called TCNN to directly transform the intensity of the speckle pattern via the multiple scattering media to the object. Compared to the traditional double phase retrieval method, this end-to-end network don’t need to model this imaging procedure and calculate the transformation matrix. Instead, it needs plenty of training data which includes the original images and corresponding intensity of speckle patterns to update the parameters of TCNN. This training procedure can be done in advance. Compared to state-of-the-arts, the recover quality of TCNN is competitive but the time TCNN cost is much less specifically recovering per image needs no more than 1s.

In TCNN, we build two sections to approximate the inverse function. The nonlinear part encodes the information of the intensity of the speckle pattern besides decoding it into a new element in transform domain, then the linear part transforms this element into the output in object domain. Test demonstrates the effectiveness of the architecture of TCNN besides tricks utilized in TCNN to alleviate the over-fit of network.

In the future, the work is to decrease the number of parameters in TCNN. Especially for the parameters in the transformation layer, they occupy large portions of the whole parameters. We
consider fuse it implicitly into the convolutional layers where the parameters are comparatively less.

5 Acknowledgement

This work was supported in part by National Natural Science foundation(China): 61571008.

References

[1] Christopher A. Metzler, Manoj K. Sharma, Sudarshan Nagesh, Richard G. Baraniuk, Oliver Cossairt, and Ashok Veeraraghavan. Coherent inverse scattering via transmission matrices: Efficient phase retrieval algorithms and a public dataset. In IEEE International Conference on Computational Photography, pages 1–16, 2017.

[2] A Velten, T Willwacher, O Gupta, A Veeraraghavan, M. G. Bawendi, and R Raskar. Recovering three-dimensional shape around a corner using ultrafast time-of-flight imaging. Nature Communications, 3(2):745, 2012.

[3] Laura Waller and Lei Tian. 3d intensity and phase imaging from light field measurements in an led array microscope. Optica, 2(2):104, 2015.

[4] I I Freund, M Rosenbluh, and S. Feng. Memory effects in propagation of optical waves through disordered media. Physical Review Letters, 61(20):2328–2331, 1988.

[5] S. M. Popoff, G Lerosey, R Carminati, M Fink, A. C. Boccara, and S Gigan. Measuring the transmission matrix in optics: an approach to the study and control of light propagation in disordered media. Physical Review Letters, 104(10):100601, 2010.

[6] M. Cui. Parallel wavefront optimization method for focusing light through random scattering media. Optics Letters, 36(6):870, 2011.

[7] Anglique Drmeau, Antoine Liutkus, Christophe Schlke, David Martina, Florent Krzakala, Laurent Daudet, Ori Katz, and Sylvain Gigan. Reference-less measurement of the transmission matrix of a highly scattering material using a dmd and phase retrieval techniques. Optics Express, 23(9):11898, 2015.

[8] Boshra Rajaei, Eric W. Tramel, Sylvain Gigan, Florent Krzakala, and Laurent Daudet. Intensity-only optical compressive imaging using a multiply scattering material and a double phase retrieval approach. In IEEE International Conference on Acoustics, Speech and Signal Processing, pages 4054–4058, 2016.

[9] R. W. Gerchberg. A practical algorithm for the determination of phase from image and diffraction plane pictures. Optik, 35:237–250, 1971.

[10] J R Fienup. Phase retrieval algorithms: a comparison. Applied Optics, 21(15):2758–2769, 1982.
Appendix
Figure 7: The paradigm of TCNN.
| Input                     | 256×256×1               |
|--------------------------|--------------------------|
| Down Sample Block        | 256×256×16               |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |
| Residue Block            | 256×256×16               |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |
| Residue Block            | 256×256×16               |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |
| Residue Block            | 256×256×16               |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |
| Residue Block            | 256×256×16               |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |
| Up Sample Block          | 256×256×16               |
|                         | 256×256×16               |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |
| Up Sample Block          | 256×256×16               |
|                         | 256×256×16               |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |
| Up Sample Block          | 256×256×16               |
|                         | 256×256×16               |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |
| Combine each flow        | 256×256×64               |
| Transformation Block     | mm×1                     |

**Down Sample Block**

| Input                     | 256×256×1               |
|--------------------------|--------------------------|
| Convolutional Layer      | 256×256×16               |
| (Kernel: 3×3×16)         |                          |
| Maxpooling Layer         | 256×256×16               |
|                         | 128×128×16               |
|                         | (Stride:2×2, Kernel:2×2) |
|                         | 64×64×16                 |
|                         | (Stride:4×4, Kernel:4×4) |
|                         | 32×32×16                 |
|                         | (Stride:8×8, Kernel:8×8) |

**Residue Block**

| Input                     | 256×256×16               |
|--------------------------|--------------------------|
| Convolutional Layer+BN+Relu | 256×256×16               |
| (Kernel:1×1×16)          |                          |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |
| Convolutional Layer+BN+Relu | 256×256×16               |
| (Kernel:1×1×16)          |                          |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |
| Short cut                | 256×256×16               |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |

**Up Sample Block**

| Input                     | 256×256×16               |
|--------------------------|--------------------------|
| Convolutional Layer+BN+Relu | 256×256×16               |
| (Kernel:1×1×16)          |                          |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |
| Decovolutional Layer     | 256×256×16               |
| (Kernel:1×1×16)          |                          |
|                         | 128×128×16               |
|                         | 64×64×16                 |

**Transformation Block**

| Input                     | 256×256×1               |
|--------------------------|--------------------------|
| Convolutional Layer      | 256×256×16               |
| (Kernel:1×1×64)          |                          |
|                         | 128×128×16               |
|                         | 64×64×16                 |
|                         | 32×32×16                 |
| Transformation Layer     | mm×1                     |

Figure 8: The details of TCNN