# Enhancing CNN-based Channel Estimation using Transfer Learning in OFDM Systems


Lingrui Zhu*, Carsten Bockelmann*, Thorsten Schier†, Salah Eddine Hajri†, Armin Dekorsy*
*Department of Communications Engineering, University of Bremen, 28359, Bremen, Germany
Email: {zhu, bockelmann, dekorsy}@ant.uni-bremen.de
†Huawei Technologies Sweden AB, 16903 Solna, Sweden
Email: {thorsten.schier, salah.eddine.hajri}@huawei.com



***Abstract*—In this paper, we investigate the application of deep transfer learning to channel estimation for an orthogonal frequency division multiplexing (OFDM) system. Recently, deep learning has been applied for channel estimation and shown its promising performance, while it suffers in the presence of a mismatch between the training phase channel model and real-world channel conditions. In the following, we deploy two different convolutional neural network (CNN) models from the literature and highlight their performance degradation caused by mismatch problems of power delay profile (PDP) and Doppler spreads. Transfer learning then is deployed to resolve the mismatch problem without the need to completely retrain CNN models. Our results show that we can alleviate the degradation using smaller efforts with transfer learning, especially for CNN with a deeper structure. In the end, when comparing transfer learning and data augmentation, our study also indicates that transfer learning is the better choice when coping with channel mismatch problems.**



***Index Terms*—deep learning, transfer learning, channel estimation, OFDM**


## I. Introduction

Orthogonal frequency division multiplexing (OFDM) has been widely employed for LTE and 5G NR to address the frequency selectivity of wireless channel and improve bandwidth efficiency. Also, channel estimation is critical for wireless communication systems. In order to obtain the channel estimates, pilot symbols will be sent at the selected time frequency grid. We can use conventional methods, i.e., Least Square (LS) and Minimum Mean Square Error (MMSE) methods to obtain a channel estimate [1]. The most significant advantage of LS is its simplicity, but the performance is also influenced by noise easily. Compared with LS, MMSE is able to achieve a better performance, however, it also requires second order channel statistics and noise variance, which must be obtained by additional estimation methods.

The recent development in Artificial Intelligence (AI) or Machine learning (ML) has reformed many areas. In [2], AI/ML is regarded as a potential technical method to improve the performance of the NR air interface. Specifically, deep learning (DL) has been investigated to enhance wireless communication systems in the areas such as channel decoding [3] and MIMO equalization [4]. For the problem of channel estimation, in [5], channel estimation for OFDM system is regarded as a Super Resolution (SR) problem and a corresponding CNN is employed. Furthermore, a special structure, residual learning, is introduced in [6] for channel estimation.

For the aforementioned deep learning based methods, a convolutional neural network (CNN) needs to be trained offline at first and then employed to estimate the wireless channel. The main disadvantage of offline learning is that, when the channel model for the employing (or testing) is different from the training phase, the estimation performance will be degraded, which can be also regraded as an distribution shift problem from the aspect of ML. However, from a practical point of view, it infeasible to collect channel data from all possible channel conditions in advance and it is very time consuming to retrain a CNN from scratch every time channel conditions change. In order to solve this problem, the concept of transfer learning [7] is applied to channel estimation in this paper, for estimating channel models different from training data. In addition to that, our investigation shows that the number of training data, epochs, and trainable layers can be reduced when deploying transfer learning. The contribution of this paper can be summarized as follows:

1. Investigate the performance degradation of CNN due to the mismatch between channel model for training phase and deploying phase.
2. Use transfer learning to resolve the performance degradation and compare the performance in two different scenarios, power delay profile (PDP) mismatch and Doppler spread mismatch.
3. Use simulation results to demonstrate the reduced number of epochs, training data, and trainable layers required for transfer learning method.

The paper is structured as follows. In section II, the system model of channel estimation is introduced. Section III explains CNN-based channel estimation methods. Next the channel mismatch problem for CNN and the concepts of transfer learning are also illustrated in this section. Simulation results are presented and discussed in IV. Finally, section V concludes the paper.

## II. System Model

The transmitter and receiver of the OFDM system are assumed to be equipped with single antennas. Each OFDM subframe consists of $N$ subcarriers and $T$ OFDM symbols.

The relationship between transmitted and received signal can be represented by

$$\mathbf{Y} = \mathbf{H} \odot \mathbf{X} + \mathbf{W} \tag{1}$$

where $\odot$ stands for the Hadamard product. $\mathbf{H} \in \mathbb{C}^{N\times T}$ represents the channel matrix containing channel coefficients. $\mathbf{X} \in \mathbb{C}^{N\times T}$ and $\mathbf{Y} \in \mathbb{C}^{N\times T}$ are transmitted signal and received signal respectively. The additive Gaussian noise is denoted as $\mathbf{W} \in \mathbb{C}^{N\times T}$. If $i$-th subcarrier and $k$-th symbol is allocated to transmit pilot symbol, it can be written as

$$\mathbf{Y}_{p,(i,k)} = \mathbf{H}_{p,(i,k)}\mathbf{X}_{p,(i,k)} + \mathbf{W}_{(i,k)} \tag{2}$$

the subscript $p$ indicates that $(i,k)$th grid is a pilot position. Channel estimation on the pilot position can be obtained using least squares:

$$\begin{aligned}\hat{\mathbf{H}}^{LS}_{p,(i,k)} &= \arg\min_{\mathbf{H}_{p,(i,k)}} \|\mathbf{Y}_{p,(i,k)} - \mathbf{H}_{p,(i,k)}\mathbf{X}_{p,(i,k)}\|^2 \\ &= \frac{\mathbf{Y}_{p,(i,k)}}{\mathbf{X}_{p,(i,k)}}\end{aligned} \tag{3}$$

### A. MMSE interpolation

In order to estimate the channel on positions where data symbols are transmitted, a linear interpolation method can be applied:

$$\hat{\mathbf{h}}_d = \mathbf{A} \cdot \hat{\mathbf{h}}_p, \tag{4}$$

where $\hat{\mathbf{h}}_p \in \mathbb{C}^{N_p}$ and $\hat{\mathbf{h}}_d \in \mathbb{C}^{N_d}$ are the vectorized channel estimates on pilot position and data position respectively. $N_p$ and $N_d$ are the number of pilot and data symbols. The interpolation matrix $\mathbf{A}$ in (4) can be calculated using the MMSE method [1]:

$$\begin{aligned}\mathbf{A}_{\text{MMSE}} &= \arg\min_{\mathbf{A}} \mathbb{E}\left\{\|\mathbf{h}_d - \mathbf{A}\cdot\hat{\mathbf{h}}_p\|^2\right\} \\ &= \mathbf{R}_{\mathbf{h}_d\mathbf{h}_p}\left(\mathbf{R}_{\mathbf{h}_p\mathbf{h}_p} + \sigma_w^2\mathbf{I}\right)^{-1}\end{aligned} \tag{5}$$

where $\mathbf{R}_{\mathbf{h}_p\mathbf{h}_p}$ is the channel auto correlation matrix at pilot positions and $\mathbf{R}_{\mathbf{h}_d\mathbf{h}_p}$ is the channel cross correlation matrix between data and pilot positions. $\sigma_w^2$ denotes noise variance and $\mathbf{I}$ represents the identity matrix. Obviously the statistics of the channel need to be perfectly known when applying MMSE interpolation, which is infeasible from a practical prospective.

### B. Radial Basis Function (RBF) Interpolation

Alternatively, kernel based approaches are also suitable for interpolation and RBF [8] is one of most commonly used kernel. Channel estimation on the data position $n$ can be calculated as:

$$\hat{\mathbf{h}}_d[n] = \sum_{j=1}^{N_p} \hat{\mathbf{h}}_p[j] \cdot \frac{\varphi_n(j)}{\sum_{l=1}^{N_p}\varphi_n(l)}, \text{for } n = 1, ..., N_d, \tag{6}$$

where $\varphi_n(l)$ is Gaussian radial basis function. The above equation can be regarded as an average with weights calculated using the kernel function $\varphi_n(\cdot)$.

## III. Deep Learning based Channel Estimation

### A. Super Resolution and CNN

Super resolution refers to the process of recovering high resolution images from low resolution images. For channel estimation, the matrix only containing channel estimates on pilot positions can be regarded as a low-resolution image and the channel estimates on both pilot and data positions as high-resolution image. In [5] and [6], neural network-based methods have been proposed to estimate the channel on data positions with the given channel estimates on pilot positions. SRCNN [5] is a shallow network which includes three convolutional layers with a structure given in Table I. Compared with SRCNN, ResNet [6] has a deeper structure and it employs residual structure to alleviate the problem of vanishing gradient for deep networks. The first layer of SR-ResNet is a convolutional layer, followed by 4 residual blocks and 2 convolutional layers at the end.

TABLE I: Network Structure for SRCNN

| Layer | Type | Kernel Num | Kernel Size | Activation |
|---|---|---|---|---|
| 1 | Conv | 64 | $9 \times 9$ | ReLU |
| 2 | Conv | 32 | $1 \times 1$ | ReLU |
| 3 | Conv | 1 | $5 \times 5$ | ReLU |

CNN based channel estimation can be divided into three steps. Firstly, obtain LS channel estimates on pilot position as explained in (3). Secondly, interpolate pilot channel estimation coarsely to estimate the channel on the data position, e.g., RBF interpolation given by (6). Thirdly, feed the coarse interpolation into CNN as input and take the output as the final estimation on both pilot and data.

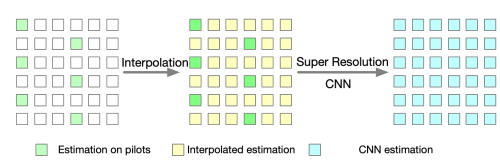



Fig. 1: The Pipeline for Deep Learning based Channel Estimation.

### B. Network Training

Denote the input of the CNN, the middle matrix in Figure 1, as $\hat{\mathbf{H}}_{\text{input}}$, the output of CNN as $\hat{\mathbf{H}}_{\text{CNN}}$. The parameters of the CNN are represented as $\Theta$ and $f_\Theta(\cdot)$ stands for the function learned by CNN. The estimation of the CNN can be written as:

$$\hat{\mathbf{H}}_{\text{CNN}} = f_\Theta(\hat{\mathbf{H}}_{\text{input}}) \tag{7}$$

Furthermore, we choose the Loss function as the Mean Square Error (MSE) between the CNN estimates and ground truth:

$$\begin{aligned}L(\Theta) &= \frac{1}{N_{\text{train}}}\sum_{l=1}^{N_{\text{train}}} \|\hat{\mathbf{H}}^{(l)}_{\text{CNN}} - \mathbf{H}^{(l)}\|^2 \\ &= \frac{1}{N_{\text{train}}}\sum_{l=1}^{N_{\text{train}}} \|f_\Theta(\hat{\mathbf{H}}^{(l)}_{\text{input}}) - \mathbf{H}^{(l)}\|^2\end{aligned} \tag{8}$$

where $N_{\text{train}}$ is the number of training data. The weights in the CNN will be updated using the gradient of the loss function with respect to the weights:

$$\Theta = \Theta - \alpha \frac{\partial L(\Theta)}{\partial \Theta} \tag{9}$$

where $\alpha$ is named the learning rate.

### C. Transfer Learning and Data Augmentation

In practice, a mismatch between the channel for training the CNN and the channel when deploying a CNN is common due to the change of environments and movements of users. If a channel changes not significantly, the CNN estimator is usually able to behave robustly [6]. However, it may be difficult for a CNN to achieve low MSE when the channel differs greatly from channel data used in training. Considering the cost, it is infeasible to train CNNs using new channel information from scratch every time the channel conditions change. In order to alleviate this problem, concepts of data augmentation and transfer learning are introduced.

*1) Transfer Learning:* The basic principle of transfer learning is to use a pre-trained model and then continuously use new data to retrain the model based on the pre-trained model. For large models, this might still be very computationally complex, but will speed up convergence of the training and requires less data generally. The most important aspect of transfer learning is based on the following observation: the front layers of the neural network will extract features containing more general information, while layers in the back close to the output will be more task-oriented [9]. Therefore, when the channel condition has changed, not all parameters need to be retrained. Front layers can be set as frozen, where the parameters would not be updated during transfer learning phase. For example, as depicted in Fig. 2, we set the first convolutional layer and subsequent three residual blocks as frozen. The corresponding parameters will not be updated when training. Data for transfer learning can be collected using channel sounding [10] or pilot aided training data generation (PATDG) [11].

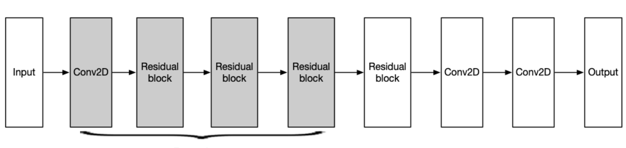



Fig. 2: An example of transfer learning for SR-ResNet. The gray block is frozen and the corresponding parameters will not be updated. The white blocks are trained.

*2) Data Augmentation:* Data augmentation is a technique commonly used in machine learning which applys transformation of existing data to increase the size and diversity of the training data [12]. In our work, data augmentation involves collecting data from different channel conditions to enhance the generality of the dataset, as changes in PDP and Doppler spread can be considered as transformations along the frequency and time axes. However, requirements on the time and computational resource in such a way for offline training will become higher due to the extra effort for data collection and the increased number of training data.

## IV. Simulation Results

### A. Simulation settings

The carrier frequency is set as 2.1 GHz and sub-carrier bandwidth as 15 kHz. The number of subcarriers $N$ is set as 72 and the number of OFDM symbols as 14. The number of pilots is 48. The modulation is 64 QAM and coding scheme is chosen as convolutional codes (with code rate of 0.5) for BER simulation. Keras and Tensorflow with GPU backend are used to build up and train CNNs.

In our work, two scenarios are taken into consideration. In scenario A, two channel models, COST 259 typical urban (CTU) and Extended Typical Urban (ETU) model are employed to evaluate estimation performance of different methods. From the PDP plotted in Fig. 3, it is clear that the ETU channel has longer delays. Therefore, the coherence bandwidth of ETU channel is narrower. For both channel models we have the same Doppler spread of 96 Hz. In scenario B, it is assumed that channel PDP is identical but user speed will change, as a result, the Doppler spread will change. The CTU channels with Doppler spreads of 96 Hz (user speed ca. 50 km/h) and 270 Hz (user speed ca. 140 km/h) are selected. In order to illustrate conveniently, these two channels are named as CTU-96 and CTU-270. In Fig. 4, different CNN training schemes are depicted for scenario A and B. The configuration for training is given by Table II. Notice that, for SRCNN, all three layers are set as trainable. However, only the last three layers in SR-ResNet are trainable and the front layers will not be updated. The SNR of training data for both initial training and transfer learning is set as 20 dB.

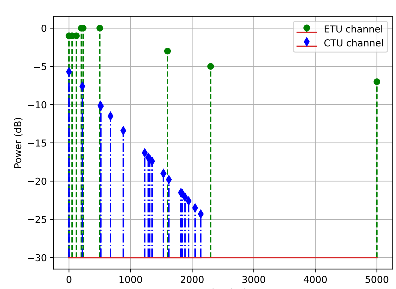



Fig. 3: Power delay profile of ETU and CTU channel

TABLE II: Configuration for initial training and transfer learning for ETU channel

| | Initial training | Transfer learning |
|---|---|---|
| Channel model | CTU / ETU 96 | ETU / ETU 270 |
| Num. training data | 10000 | 3000 |
| Num. training epochs | 100 | 10 |

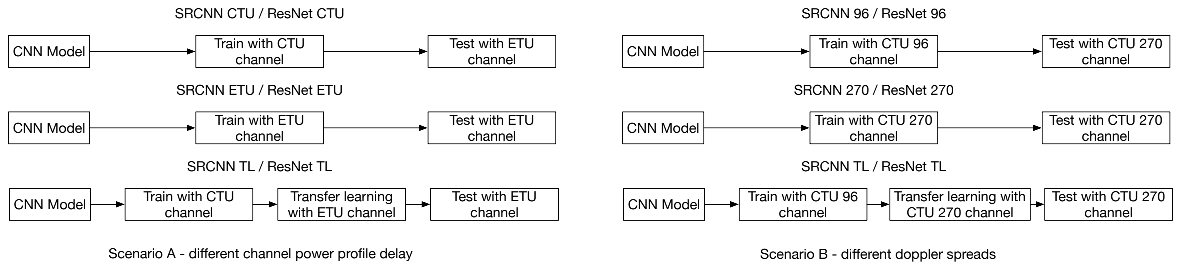



Fig. 4: Training process for different CNN based methods

TABLE III: Parameter settings for BER simulation

| | |
|---|---|
| Coding Scheme | Convolutional codes |
| Code rate | 0.5 |
| Modulation | 64 QAM |
| Channel Model | ETU Channel |
| Channel Equalization | Least Square |

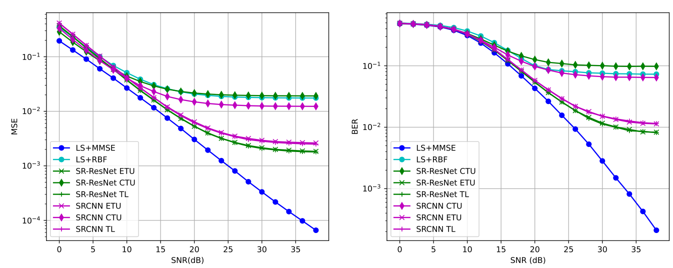
Besides the CNN methods, we also simulated other algorithms for comparison.

- LS+RBF means that we at first get the LS estimation on pilot position and then use RBF to interpolate and obtain the estimation also on the data position, as given in (6).
- LS+MMSE stands for the method that LS estimations on pilot position are interpolated using MMSE method, as illustrated in (4) and (5). Channel statistics and noise variances are assumed to be known perfectly, which is not feasible from a practical standpoint.

## *B. Performance analysis*

*1) Scenario A:* The estimation performance for the ETU channel is depicted in Fig. 5a. Due to the assumed perfect knowledge of channel statistics and noise variances for MMSE interpolation, LS+MMSE exhibits the best performance. While LS+RBF has the highest MSE until approximately 22 dB and after that SR-ResNet CTU has the highest MSE. CTU trained models (SRCNN CTU and SR-ResNet CTU) have mismatched training and testing data, therefore it can be observed that gaps between CTU trained models and ETU trained or transfer learning model are significant when SNR is greater than 15 dB. Interestingly, we can find that ResNet model is more sensitive to the channel mismatch since the degradation of SR-ResNet is more significant than SRCNN. The reason is that SR-ResNet has a larger number of parameters and a deeper structure, so it can overfit to the training data to a larger degree. The overlapping between ETU trained models and TL methods proves that transfer learning is able to alleviate this problem to a large degree for both SRCNN and SR-ResNet.

The comparison of bit error rates (BER) is plotted in Fig. 5b. We use the estimates of introduced methods to equalize the channel and the corresponding simulation settings is given in table III. Similarly, the degradation due to channel mismatch is also significant and transfer learning is helpful to solve this problem. It is interesting that the difference between LS+RBF and SR-ResNet CTU is larger than that shown in Fig. 5a. The reason is that LS+RBF has some extremely large estimation errors for certain data symbols. The large value of a single error can increase the MSE to a large degree but it has limited influence on the error rate simulation, since the maximum contribution cannot be more than one symbol error.




(a) MSE (b) BER

Fig. 5: Scenario A: Channel Estimation MSE and BER for ETU channel in terms of SNR.

In Figure 6, we demonstrate how the computational resources required for transfer learning with SR-ResNet can be further reduced in terms of three aspects: the number of epochs, training data, and trainable layers. The default settings is as following: 10 epochs, 3000 training data and 3 trainable layers. For each plot, we just change one setting and keep the other two defaulting settings. From (a), we can find that 3 training epochs for transfer learning is already able to reduce MSE to a relatively low level. Similarly, from part (b) and (c), 3000 training data and 1 trainable layers can bring improvements to a large degree. Compared with training a model from scratch, the cost of transfer learning is quite acceptable due to the small number of trainable layer, training data and epochs.

*2) Scenario B:* In Fig. 7a, estimation MSEs for ETU-270 channel are plotted. The gap between SR-ResNet-96/SRCNN-96 and SR-ResNet-270/SRCNN-270 shows that the mismatch due to different Doppler spread can also cause performance degradation. TL models perform much better than SR-ResNet-96/SRCNN-96, but worse than SR-ResNet-270/SRCNN-270. A possible explanation is that difference between Doppler spread 270 Hz and 96 Hz is large and it needs more training data or epochs to alleviate completely.

Fig. 7b provides the comparison of BERs of CNN-based and benchmark methods. For SNRs below 10 dB, BER differences are insignificant. When SNRs are higher, especially greater

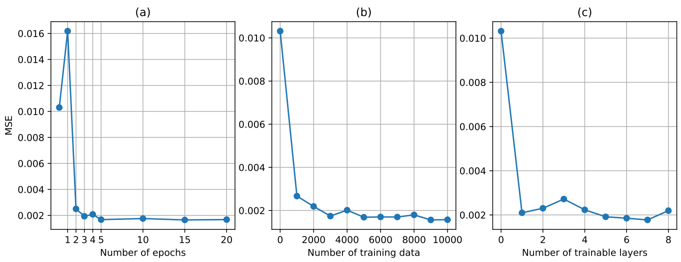
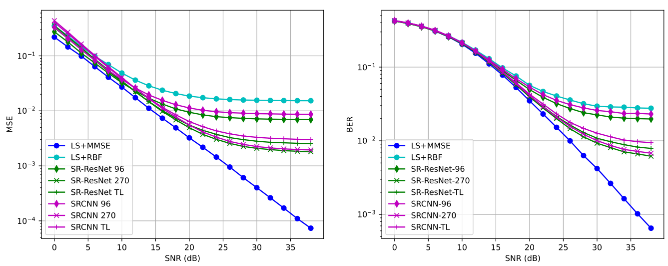



Fig. 6: Channel Estimation MSE of transfer learning in terms of (a) number of epochs, (b) number of training data, and (c) number of trainable layers.




(a) MSE (b) BER

Fig. 7: Scenario B: Channel Estimation MSE and BER for CTU channel (Doppler spread 270 Hz) in terms of SNR.

than 20 dB, it is very clear that LS+MMSE has the best performance and LS+RBF has the worst one. Transfer learning based methods outperform mismatched trained models (i.e., SR-ResNet-96/SRCNN-96), but are inferior to matched trained models (i.e., SR-ResNet-270/SRCNN-270). From the comparison depicted in Fig. 7a and 7b, it is evident that the order of channel estimation MSE is consistent with the order of BER.

The comparison between data augmentation and transfer learning is given in Figure 8. At first, 10000 data samples from CTU-96 and 3000 data samples CTU-270 channel are collected in advance to build up a dataset with higher generality. SRCNN-DA/SR-ResNet-DA are trained using this new general dataset. For SNRs below 10dB, SR-ResNet models are better than SRCNN models regardlesss of the training strategy. For the same CNN model, we can find TL strategy is able to have a lower MSE. In high-SNR region, the differences between DA and TL are more significant. Above observations indicate that TL is the better solution when solving the data mismatch problem in the context of channel estimation.

Overall, the MSE and BER performance have been improved by transfer learning with a relative low cost for the two simulation scenarios, PDP and Doppler spread mismatch. In the end, we also found that transfer learning is able to perform better than data augmentation when coping with the channel model mismatch problem.

## V. Conclusion

In this paper, we applied deep transfer learning to channel estimation. When the channel for training and channel for inference phase are mismatched because of different PDPs

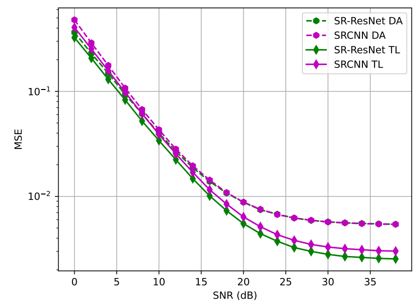



Fig. 8: Comparison of channel estimation MSE for models trained using transfer learning and data augmentation.

or Doppler spreads, transfer learning can alleviate the performance degradation using a small effort. We have shown that transfer learning can increase the scalability and flexibility of the CNN based channel estimation method at low cost. Furthermore, the comparison between transfer learning and data augmentation also indicated that transfer learning is the superior approach when dealing with channel mismatches.

## References

[1] Xiaodai Dong, Wu-Sheng Lu, and Anthony CK Soong. Linear interpolation in pilot symbol assisted channel estimation for ofdm. *IEEE transactions on wireless communications*, 6(5):1910–1920, 2007.

[2] 3GPP. SID on AI/ML for NR Air Interface . Work item description, 3rd Generation Partnership Project (3GPP), 7 2021. RP-221348.

[3] Tobias Gruber, Sebastian Cammerer, Jakob Hoydis, and Stephan ten Brink. On deep learning-based channel decoding. In *2017 51st annual conference on information sciences and systems (CISS)*, pages 1–6. IEEE, 2017.

[4] Aldebaro Klautau, Nuria González-Prelcic, Amine Mezghani, and Robert W Heath. Detection and channel equalization with deep learning for low resolution mimo systems. In *2018 52nd Asilomar conference on signals, systems, and computers*, pages 1836–1840. IEEE, 2018.

[5] Mehran Soltani, Vahid Pourahmadi, Ali Mirzaei, and Hamid Sheikhzadeh. Deep learning-based channel estimation. *IEEE Communications Letters*, 23(4):652–655, 2019.

[6] Lianjun Li, Hao Chen, Hao-Hsuan Chang, and Lingjia Liu. Deep residual learning meets ofdm channel estimation. *IEEE Wireless Communications Letters*, 9(5):615–618, 2019.

[7] Ricardo Ribani and Mauricio Marengoni. A survey of transfer learning for convolutional neural networks. In *2019 32nd SIBGRAPI conference on graphics, patterns and images tutorials (SIBGRAPI-T)*, pages 47–57. IEEE, 2019.

[8] M Hoseinzade, K Mohamedpour, SMH Andargoli, and H Shokri Razaghi. Ofdm channel estimation based on gaussian radial basis function interpolation. In *2009 11th International Conference on Advanced Communication Technology*, volume 1, pages 9–13. IEEE, 2009.

[9] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014.

[10] Tao Zhou, Cheng Tao, Sana Salous, Liu Liu, and Zhenhui Tan. Channel sounding for high-speed railway communication systems. *IEEE Communications Magazine*, 53(10):70–77, 2015.

[11] Kai Mei, Jun Liu, Xiaoying Zhang, Kuo Cao, Nandana Rajatheva, and Jibo Wei. A low complexity learning-based channel estimation for ofdm systems with online training. *IEEE Transactions on Communications*, 69(10):6722–6733, 2021.

[12] Andrei Dmitri Gavrilov, Alex Jordache, Maya Vasdani, and Jack Deng. Preventing model overfitting and underfitting in convolutional neural networks. *International Journal of Software Science and Computational Intelligence (IJSSCI)*, 10(4):19–28, 2018.