Turkish Journal of Electrical Engineering and Computer Sciences

Volume 21 | Number 5 Article 18

1-1-2013

A learning approach in link adaptation for MIMO-OFDM systems

HALİL YİĞİT

ADNAN KAVAK

Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2013) 21: 1465 – 1478

doi:10.3906/elk-1110-24

A learning approach in link adaptation for MIMO-OFDM systems

Halil YİĞİT,[1,2,*] **Adnan KAVAK**[2,3]

[1]Department of Information Systems Engineering, Technology Faculty, Kocaeli University, İzmit, Kocaeli, Turkey
[2]Wireless Communications & Information Systems Research Center, Kocaeli University, İzmit, Kocaeli, Turkey
[3]Department of Computer Engineering, Faculty of Engineering, Kocaeli University, İzmit, Kocaeli, Turkey

Received: 12.10.2011 • **Accepted:** 03.05.2012 • **Published Online:** 12.08.2013 • **Printed:** 06.09.2013

Abstract: We propose a neural network (NN)-based adaptive modulation and coding (AMC) for link adaptation in MIMO-OFDM systems. The AMC optimizes the best modulation and coding scheme (MCS) under a packet error rate (PER) constraint. In our approach, a NN with a multilayer perceptron (MLP) structure is applied for the AMC and its performance is compared with the k-nearest neighbor (k-NN) algorithm under the frequency-flat (1-tap) and frequency-selective (4-tap) wireless channel conditions. The simulation results show that the NN classifier outperforms the k-NN algorithm, especially in terms of the PER, due to the fact that the MLP guarantees a MCS with a lower data rate by way of the selection of a class label with a lower index number. It has a slightly worse spectral efficiency performance compared to the k-NN. Thus, the MLP approach provides higher communication robustness over the k-NN. It can be concluded from the results that the selection of the AMC classifier depends on a trade-off between the PER and the spectral efficiency, relying on the user's requirements.

Key words: MIMO systems, link adaptation, adaptive systems, neural nets, learning systems

1. Introduction

The performance of multiple-input multiple-output orthogonal frequency division multiplexing (MIMO-OFDM) can be increased using a link adaptation in the form of adaptive modulation and coding (AMC) [1–3]. The AMC in MIMO-OFDM selects the best modulation order and coding rate parameters based on a link quality metric (LQM) obtained from the channel state information (CSI) to reduce the packet error rate (PER) and maximize the data rate under the PER reliability constraint [4,5].

In MIMO-OFDM systems, there are various difficulties in the implementation of the AMC [6]. The delayed CSI feedback at the transmitter by reason of the effect of the transmission delays and errors in the feedback causes a significant decrease in the AMC's performance [7]. An important factor to consider for the AMC is the quality of the PER prediction. To predict the PER, a general solution is to generate a look-up table (LUT) for a LQM, which includes all of the effects in wireless channel $\mathbf{H}$ between the transmitter and the receiver, noise power N_0, transmitted signal energy E_s, packet length in bits L, and the modulation and coding scheme (MCS) selected [8,9]. The most important disadvantage of using LUTs is that the memory usage is significant in high-dimensional LQMs, and so the LUT is generally restricted to 1- or 2-dimensional LQMs for simplicity [10]. In addition, LUTs rely seriously on the wireless channel model and the unique traits of each wireless device. There are a variety of LQMs for the AMC. However, the most commonly used LQM

*Correspondence: halilyigit@kocaeli.edu.tr

is the postprocessing signal-to-noise ratio (SNR) defined for each spatial stream and each subcarrier in the MIMO-OFDM system [10,11].

In this paper, we propose a neural network (NN) classifier as a machine learning (ML) technique for the AMC in the MIMO-OFDM system under the frequency-flat channels (FFCs) and the frequency-selective channels (FSCs). The advantages of NNs include the requirement of less statistical training, the ability to determine nonlinear statistical relationships between the inputs and outputs, good generalization capability, and the availability of various learning algorithms. Once a NN is trained with the training data, mapping between the inputs and outputs occurs very fast in the testing stage. Furthermore, a NN can be used effectively, even when the dimension of its input and output is high. The main drawback of the NNs is that they include a computationally intensive training process to minimize overfitting [12]. As the training operation does not need to be performed frequently, this is not a vital shortcoming.

As a NN classifier, we consider a multilayer perceptron (MLP), which is one of the most popular NN models, to detect the relationships between the LQM and the MCS index, which includes the modulation order and coding rate parameter for the AMC in MIMO-OFDM systems. The Levenberg–Marquardt (LM) algorithm [12,13], which is the fastest backpropagation algorithm in terms of convergence to the optimum point, is used for training the proposed MLP network.

We compare our proposed NN structure with the k-nearest neighbor (k-NN) algorithm since both use supervised learning. However, the k-NN algorithm does not require any training process in contrast to the NN. In the literature, it is known as a lazy classifier due to the fact that models are not built explicitly, unlike the NN and other classification algorithms. The k-NN algorithm requires the computation of the distance of the unlabeled object to all of the labeled objects. Furthermore, the selection of the k value significantly affects the performance of the k-NN algorithm [13,14].

1.1. Related works

The application areas for NNs in wireless communications are still attractive for researchers. The authors of [15] carried out channel prediction in order to track the mobile station position and to increase the received signal power in varying wireless channels using linear NNs. In [11,16], the k-NN algorithm was implemented to provide an accurate MCS using different LQMs. A narrow-scoped study for AMC based on a NN approach was performed due to the paper's length limitation in [17], where the FSC is only studied with a lack of computational complexity. In [18,19], the support vector machine (SVM) algorithm was applied to optimize the AMC parameters in MIMO-OFDM systems. The authors compared the PER and the average throughput performance of the SVM and k-NN classifier. Although SVMs have significant advantages over NNs, especially with regard to finding the global minimum, they are relatively slow in the test stage. SVMs have high algorithmic complexity and extensive memory requirements both in training and testing. Contrary to the SVM, the input-output mapping is computationally very light in NNs and, therefore, it is considered as well matched for real-time systems. An important practical problem that has not been solved yet for SVMs is the selection of the kernel function parameters [20]. In NNs, it is only necessary to find the layer and neuron number. With the motivation of the fast input-output mapping capability of NNs compared to k-NN and SVM, our aim in this study is to demonstrate that the NN approach is applicable for AMC in a MIMO-OFDM system.

The remainder of this paper is organized as follows. Section 2 describes the MIMO-OFDM system model. In Section 3, an overview of the AMC for link adaptation is given. The classification approaches for the AMC in MIMO-OFDM are presented in Section 4. The detailed simulation setup and results are given in Section 5. The concluding remarks are stated in Section 6.

2. System model

The MIMO-OFDM system model and IEEE 802.11n physical layer specifications [21] are employed for the MCS selection described in this study. The MIMO-OFDM system is illustrated in Figure 1.

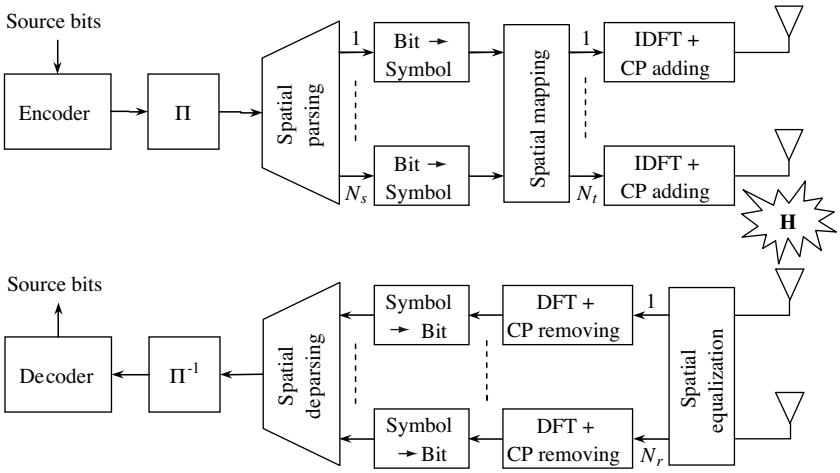

Figure 1. Block diagram of the MIMO-OFDM wireless communication system.

At the transmitter, the source bits are encoded with the convolutional coding rate C. The coded and interleaved bits are parsed $N_s \in \min(N_r, N_t)$ spatial streams, where N_r and N_t are the number of receive antenna and transmit antenna, respectively. Each spatial stream is independently mapped into symbols using the quadrature amplitude modulation (QAM) constellation order M. The spatial mapping converts N_s dimensions into N_t dimensions. The time-domain signals are transmitted over N subcarriers using the inverse discrete Fourier transform (IDFT) algorithm and cyclic prefix (CP) adding. At the receiver, the N_r-dimensional time-domain signals are first received. Spatial equalization is performed to obtain the estimate of the received signals. The frequency domain OFDM symbols given by Eq. (1) are obtained by implementing the discrete Fourier transform (DFT) and CP, removing the equalized samples. After that, the QAM symbols are mapped to the bits per spatial branch. Each branch is coupled together via a deparsing operation. After deinterleaving, the receiver obtains the final binary sequence using a Viterbi decoder [10]. For an MIMO-OFDM system equipped with N_t transmit antennas and N_r receive antennas, the received symbol on the nth subcarrier $n \in \{1,2,\ldots,N\}$ after DFT is given by:

$$\mathbf{y}_n = \sqrt{E_s}\mathbf{H}_n\mathbf{x}_n + \mathbf{v}_n, \tag{1}$$

where $\mathbf{x}_n$ is the transmit symbol vector; $\mathbf{v}_n \sim CN(0,\ N_0\mathbf{I})$ is the additive complex Gaussian noise vector, where the real and imaginary components are independent and identically distributed (IID) normal random variables with a zero mean and variance N_0; and E_s is the expected total transmit energy. $\mathbf{H}_n$ represents the channel matrix of the nth subcarrier. The channel coefficients in the frequency domain are obtained as linear combinations of the dispersive channel taps. It is assumed that the channel impulse response is static for all of the OFDM symbols. The expected transmit energy E_s and noise variance N_0 are normalized to 1 to characterize the input-output relationship of the MIMO-OFDM system.

3. Link adaptation based on AMC

The basic idea of link adaptation based on AMC is to select the optimal AMC parameters, such as the modulation order M and convolutional coding rate C, to maximize the throughput R under the PER constraint as given by:

$$\arg\max_{i} \{R_i : PER_i \leq PER_{t\arg et}\}, \tag{2}$$

where the index i corresponds to a MCS defined by the IEEE 802.11n physical layer specifications [21]. In this study, the index i is the class label of the MCS in terms of the classification. In this case, Eq. (2) states that a class i is selected if the corresponding PER_i is less than or equal to the target PER for a given channel realization $\mathbf{H}$. The PER is a function given by:

$$PER_i = f\{MCS_i, \mathbf{H}, E_s, N_0, L\}, \tag{3}$$

where L is the packet length in bits. The spectral efficiency of the MIMO-OFDM system for a given channel realization is defined as the data rate of the optimum MCS divided by the channel bandwidth B. It is mathematically given by:

$$\zeta_i = R_i\,(1 - PER_i(\gamma))/B \ \text{ bps/Hz}, \tag{4}$$

where γ is the postprocessing SNR per spatial stream and per subcarrier, which is defined as a LQM to describe the performance of the link adaptation. At the receiver side, if the linear zero-forcing (ZF) equalizer is used, the ZF postprocessing SNR on the each spatial stream $a \in \{1,\ldots,N_s\}$ and each subcarrier $n \in \{1,\ldots,N\}$ with agreeable Gaussian interstream interference plus noise is expressed as:

$$\gamma[a,n] = \frac{E_s}{N_0 \sum\limits_{a'=1}^{N_s} \left|[\mathbf{G}_{ZF}[n]]_{a,a'}\right|^2}, \tag{5}$$

where $\mathbf{G}_{ZF} = (\mathbf{H}_n)^{+} = (\mathbf{H}_n^H\mathbf{H}_n)^{-1}\mathbf{H}_n^H.(\cdot)^H$ and $(\cdot)^{-1}$ denote the Hermitian transpose and inverse matrix, respectively. $\mathbf{G}_{ZF}$ is the Moore–Penrose pseudoinverse matrix of $\mathbf{H}$ for the nth subcarrier [11].

4. Machine learning approaches

In this section, we apply the k-NN algorithm and the MLP network to the AMC in a MIMO-OFDM system. Both classifiers exploit the supervised learning algorithm to select the correct class labels based on the training set. To perform the classification, a training set of feature sets and correct class labels indexed by a nonnegative integer is created. For our study, each element of the feature set $\mathbf{z}$ corresponds to the postprocessing SNR values of each spatial stream $a \in \{1,\ldots,N_s\}$ and each subcarrier $n \in \{1,\ldots,N\}$ calculated according to Eq. (5), and each class label i corresponds to a MCS. To obtain the training set, each distinct realization of the feature set is assigned to a class label i according to Eq. (2). Mathematically, it is defined for each channel realization as follows:

$$\{\mathbf{z}_w\} \Rightarrow \{i(w)\}, \tag{6}$$

$$\mathbf{z}_w = \left[\{\gamma_w[a,n]\}_{a=1,n=1}^{N_s,N}\right] \tag{7}$$

where w is a realization index for each feature set in the training set.

4.1. AMC based on the k-NN algorithm

The k-NN algorithm can select the class labels accurately without using functional mapping between the feature sets $\mathbf{z}$ and the class labels i. Since it does not employ a training process, the k-NN algorithm searches the entire training set using the Euclidean distance metric for a query given from a channel realization. After searching the operation, an ascending sort is performed over the distances. Once the k-NN list is obtained, the query is classified based on the majority vote (weighted equally) of its nearest neighbors [14]. Figure 2 shows the majority vote principle in the k-NN classification. In this instance, the query (black-filled square) has 2 neighborhoods to the MCS_9 $(i = 9)$ class, 3 neighborhoods to the MCS_{10} $(i = 10)$ class, and 1 neighborhood to the MCS_{12} $(i = 12)$ class, respectively. Thus, MCS_{10} is selected considering that the query has a larger number of neighborhoods to MCS_{10} than to either MCS_9 or MCS_{12}. On the other hand, if 2 or more classes result in the same number of neighbors to the query, a class label with a lower index is selected. It is clearly seen from Figure 3 that the query has 3 neighborhoods to the MCS_9 and MCS_{10} classes, respectively. In this case, MCS_9 is selected for the AMC. The unique difference from [11,18] is that a class label indexed by a nonnegative integer number is the only feedback to the transmitter. The transmitter sends information bits at the maximum data rate using modulation order M and coding rate C selected from Table 1.

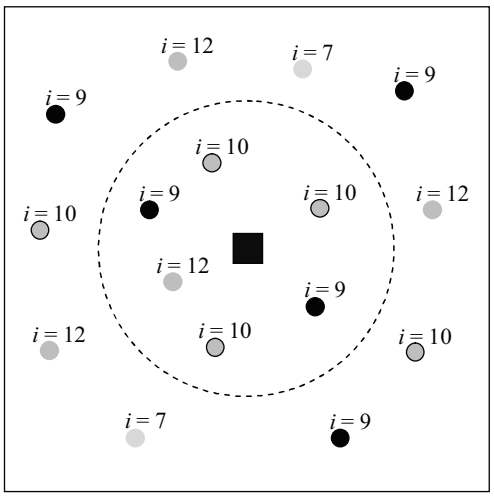

Figure 2. MCS class selection based on a majority vote.

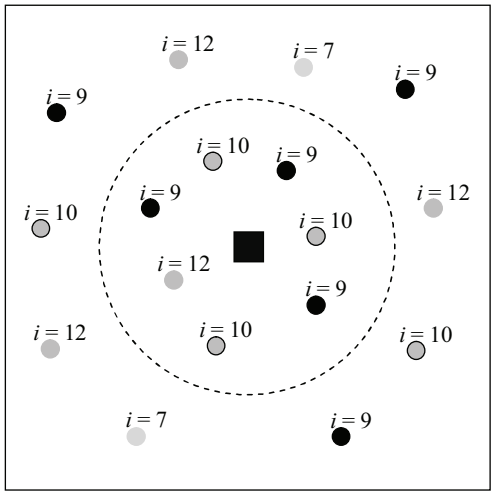

Figure 3. The presentation of the MCS_9 selection in a 6-NN classifier.

Table 1. IEEE 802.11n MCS list for the spatial multiplexing mode.

MCS index value	M	C	R_i
$i = 8$	2	1/2	13.0 Mbps
$i = 9$	4	1/2	26.0 Mbps
$i = 10$	4	3/4	39.0 Mbps
$i = 11$	16	1/2	52.0 Mbps
$i = 12$	16	3/4	78.0 Mbps
$i = 13$	64	2/3	104.0 Mbps
$i = 14$	64	3/4	117.0 Mbps
$i = 15$	64	5/6	130.0 Mbps

4.2. AMC based on the MLP network

Contrary to the k-NN algorithm, the MLP network performs functional mapping between its inputs and outputs. Figure 4 illustrates a MLP network model with a single neuron. The input to this neuron, $\mathbf{x} = (x_1, x_2, \ldots, x_p)$, is a feature vector in a p-dimensional feature space. $v_j \{1 \leq j \leq p\}$ are the synaptic weights. The quantity b is called the bias.

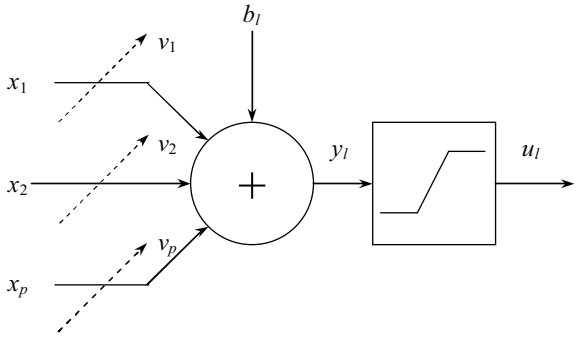

Figure 4. MLP network with a single neuron.

In Figure 4, the network output u_l is related to the network input y_l via a linear or nonlinear activation function as given by Eq. (8):

$$u_l = f\left(\sum_{j=1}^{p} v_j x_j + b_l\right) = f\left(y_l\right), \tag{8}$$

where l is the neuron index in the case that multiple neurons are used in the MLP. The tangent-sigmoid function is usually exploited as a nonlinear activation function. It is mathematically given as follows:

$$f(y) = \frac{2}{1 + \exp(-2y)} - 1. \tag{9}$$

In the linear activation function, the network output directly equals the network input y, as given by Eq. (10).

$$f(y) = \sum_{j=1}^{p} v_j x_j + b \tag{10}$$

During the training process, the feature vector $\mathbf{x}$ is fed to the network. After nonlinear/linear mapping, the network output u is obtained. The weights in the network are updated according to the chosen learning algorithm to minimize the error e between every output value u and its target value u_d, as shown in Figure 5. The training is stopped when the weights converge to a particular solution for the training set.

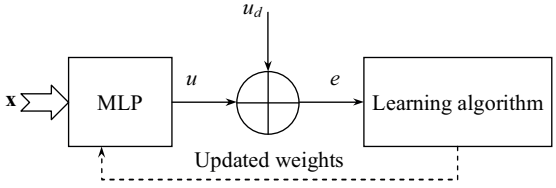

Figure 5. The training process in the MLP network.

The input-output relationship for the MLP AMC can be expressed as follows:

$$i(w) = \Theta\left(\mathbf{z}_w\right), \tag{11}$$

where $\mathbf{z}$ is the feature set and each column has a p-dimensional feature vector $\mathbf{x}$, and i is the network output that corresponds to the class label. $\Theta(\cdot)$ is the total function that represents the NN model.

We have chosen a MLP structure, as depicted in Figure 6. Before feeding the training data, which contain the feature set and class labels, to the network, we normalize to the range of –1 and +1 in stage 1. In stage 2, an overall function of the MLP network is obtained to perform a mapping between its input and output according to the predefined network parameters. Stage 3 performs an inverse normalization operation to obtain a NN output before normalization. The network output might be noninteger after the reverse normalization. At this point, the crucial property of the MLP-based model is a rounding operation. Stage 4 rounds this value to 2 integers. The first value corresponds to the nearest integer towards the infinity of the network output (i.e. $7.2 \rightarrow 8$). The other is the nearest integer towards zero (i.e. $7.2 \rightarrow 7$). After that, we choose the class label i, which both stays within the MCS set illustrated in Table 1 and has a lower class label number. For example, if the network output rounds to 7 and 8 after the rounding operations, the class label i is selected as 8. Because the MCS set does not allow us to use MCS_7 in the case of 2 spatial streams, $N_s = 2$ in the IEEE 802.11n MIMO-OFDM system. Thus, we ensure the selection of the MCS with a low class label number, which provides higher communication robustness. The reason for choosing only 2 spatial streams is that most of the errors occur in the $N_s = 2$ condition.

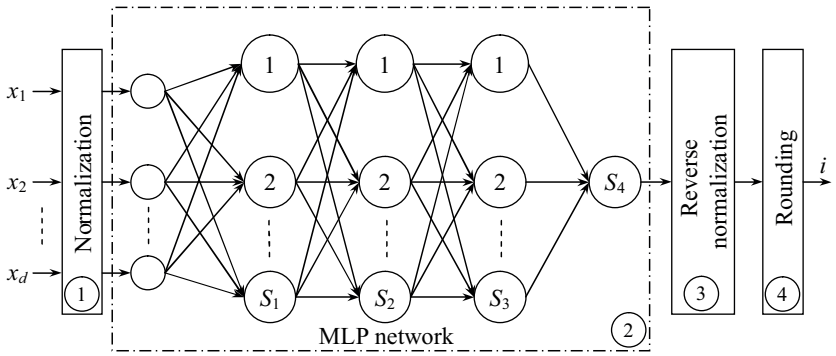

Figure 6. MLP network structure for the AMC classification.

5. Simulation setup and results

5.1. Simulation setup

The IEEE 802.11n MIMO-OFDM system with 2 receive antennas and 2 transmit antennas is considered to evaluate the performance of the proposed AMC classification. We assume a MCS set for the spatial multiplexing mode ($N_s = 2$) depicted in Table 1.

The MIMO-OFDM system is simulated for each modulation order M, coding rate C, and the 2 spatial streams to determine the PER in the FFCs (1-tap) and in the FSCs with 4 taps, respectively. For the simulations, the system assumes that the data length is 128 bytes per packet. The number of data subcarriers N is 52. The FFT size in the IEEE 802.11n is selected as 64 for 20 MHz channels. Perfect frequency synchronization and symbol timing is assumed. The receiver has perfect channel state information. The soft-output ZF spatial equalization is exploited at the receiver. Neither space-time block coding nor precoding occurs during the simulations.

In the simulations, we use the previously generated channel data set #1 for the training phase and channel data set #2 for the testing phase. In both channel data sets, each entry of the data set has a $N_r \times N_{taps} \times N_t$ channel matrix, where N_{taps} is the number of channel taps. Each element of the channel matrix is an IID complex-Gaussian random variable with a zero mean and variance N_0. The number of channel realizations is 3.2×10^4. The system is run for each MCS in one channel realization. Hence, the PER vectors are also 3.2×10^4-dimensional for each MCS. We have 8 PER values, each corresponding to a distinct MCS for a given channel realization.

To create the training set, we use 3.2×10^4 channel realizations of channel data set #1. For each channel realization w, the postprocessing SNR values γ are computed according to Eq. (5) and sorted in ascending order. The N_sN-dimensional ordered postprocessing SNR set is obtained. We then select the best indices p of these sorted SNR values, which gives an accurate approximation of the N_sN-dimensional set, to reduce the feature set dimension. Although the NN and the k-NN classifier do not require the reduction of the feature set dimension, it might be beneficial to reduce the computational complexity, severe memory usage, and classification time. The best indices for the ordered postprocessing SNR sets are determined via a brute force search (BFS) over all of the feature sets [10]. Table 2 shows the best subcarrier indices up to 4 dimensions, defined by the BFS for the IEEE 802.11n MIMO-OFDM system in the FSCs (4-tap). In the FFC, the postprocessing SNR values for each spatial stream and subcarrier remain the same. Consequently, there is no need to determine the indices for the 1-tap channel.

Table 2. The best subcarrier indices.

N_s	p	4 taps
2	1	{25}
2	2	{11, 52}
2	3	{7, 24, 61}
2	4	{6, 13, 24, 56}

In Table 2, the index numbers of more than 4 are not included. This is because $p > 4$ does not have a significant performance improvement over $p = 4$ [11]. As a consequence, to construct the 4-dimensional feature vector $\mathbf{z}$, we choose the indices as

$$\mathbf{z} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \gamma_6 \\ \gamma_{13} \\ \gamma_{24} \\ \gamma_{56} \end{bmatrix} \tag{12}$$

for each channel realization.

After the dimension reduction, we construct the training set in the NN-based AMC. For this purpose, a new feature set is assigned to the correct class label in the training set according to Eq. (2). The target PER is chosen as 0.1 (10% PER). After determining the MCS_i set that meets the PER constraint for a given channel, the class label i with the highest rate R is chosen. If the PER constraint cannot be met by any class, the MCS_i with the lowest rate R is selected. Table 3 shows the training data set example for the AMC where K is the total number of entries in the training set. Each row in Table 3 consists of the p-dimensional feature vector, where each element represents the postprocessing SNR value as the network input and class label i ranging from {8–15} as the network output.

After the training process, the network is then verified on 3.2×10^4 channel realizations within channel data set #2. As mentioned in Section 4.2, the selection of the class label with a lower rate is ensured from stage

3 in Figure 6. The PER values and spectral efficiencies obtained for all of the channel realizations are plotted. The postprocessing SNR values used as the LQM are averaged over all of the subcarriers and spatial streams. Next, a number of average LQM bins are created and the PERs and the spectral efficiencies in each bin are averaged, as depicted in Figure 7.

Table 3. The training set example for the AMC.

Feature sets $\mathbf{z}$				Class labels i
γ_{11}	γ_{12}	...	γ_{1p}	8
γ_{21}	γ_{22}	...	γ_{2p}	15
...	...	...	...	...
γ_{K1}	γ_{K2}	...	γ_{Kp}	12

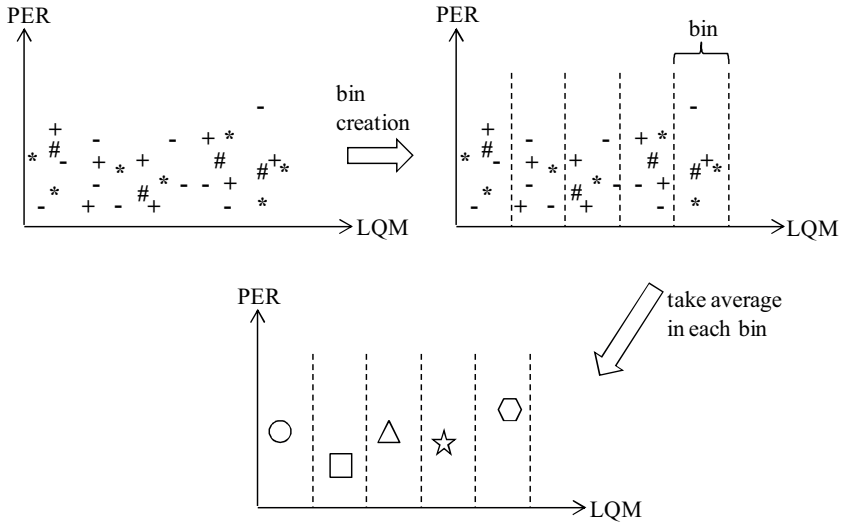

Figure 7. The principle of defining bins.

5.2. Complexity analysis

In this section, we focus on the computational complexity of selecting the optimum MCS index for the k-NN and MLP classifiers. The k-NN algorithm basically involves Euclidean distance, which is the d computation between 2 p-dimensional vectors, as follows:

$$d^2(\mathbf{a}, \mathbf{b}) = \sum_{j=1}^{p} (a_j - b_j)^2. \tag{13}$$

This results in pK multiplies and $2pK$ additions, where K is the distance computation number, or, in other words, the number of feature sets in the training set. In practical implementations, the square of the distance is used instead of the distance to get rid of the square root operation. It does not affect the performance of the k-NN algorithm. After calculating the square of the distances, a K-length sorting operation is required. To determine the nearest neighbors with a majority vote, the class labels are assigned to K sorted distances via a comparing operation that requires $\log_2(K)$ computations and k + 1 additions. The test set contains K feature sets. Finally, the total computational complexity of the k-NN algorithm is found by multiplying all of the computation numbers with K.

In the MLP approach, we consider a network with 2 hidden layers that contain 5 neurons each. As mentioned above, to train the MLP network, LM optimization is used. The weight update expression of the LM algorithm can be given as follows:

$$\mathbf{W} = \mathbf{W} - \left[\mathbf{J}^T\mathbf{J} + \mu\mathbf{I}\right]^{-1}\mathbf{J}^T\mathbf{e}, \tag{14}$$

where $\mathbf{J}$ is the Jacobian matrix and $\mathbf{e}$ is the error vector for all of the inputs. LM requires the computation of the $\mathbf{J}$ matrix, as seen from Eq. (14). We suppose that $\mathbf{J}$ is known. In our case of K patterns, and the MLP network with a total of 61 weights and 1 output, the Jacobian matrix has K rows and $W = 61$ columns. The $\mathbf{J}$ matrix becomes $K \times W$-dimensional. The computation of $\mathbf{J}^T\mathbf{J}$ requires KW^2 multiplications and KW^2 additions. To calculate the $\mu\mathbf{I}$ production, W^2 multiplications is needed, where $\mathbf{I}$ is the $W \times W$-dimensional identity matrix. The sum of 2 matrices of $W \times W$ size requires $2W^2$ additions. After the sum operation, the matrix inversion is performed for the $W \times W$ matrix. Here, the singular value decomposition (SVD) algorithm is implemented to achieve an inverse operation. In [15], the authors performed a computational complexity analysis of the SVD algorithm. This paper does not supply the derivation of the SVD-based inverse operation. It can be shown that it contains $29W^3 + 17W^2 + 4W - 42$ additions, $29W^3 + 17W^2 + 2W - 42$ multiplications, and $\log_2(W)$ comparison. To calculate the $\mathbf{e}$ vector for all of the K patterns, $2K$ additions are performed. $\mathbf{J}^T\mathbf{e}$ requires KW additions and multiplications. The product of 2 matrices of size $W \times W$ and $W \times 1$ has W^2 additions and W^2 multiplications. Two $W \times 1$-dimensional vectors are finally subtracted to determine all of the weights. This operation requires only $2W$ additions. After determining all of the weights, to obtain i, the MLP network performs $f(\mathbf{W}^T\mathbf{X} + \mathbf{b})$ mapping over all of the layers and neurons, where f is the activation function given by Eq. (9), $\mathbf{X}$ is the inputs to the neurons, and $\mathbf{b}$ is the bias vector. The mapping between all of the inputs and outputs requires $(5p + 61)$ K multiplications, $(5p + 61)$ K additions, and 10 exponential operations per epoch. Table 4 gives a summary of the computational complexity analysis of the k-NN algorithm and the MLP network to predict the best MCS class index for all of the K test patterns.

Table 4. A summary of the computational complexity analysis of the k-NN and MLP classifiers.

k-NN	Addition	Multiplication	Sort	Comp.	Exp.
Train	...	...	...	...	...
Test	$2pK^2 + (k + 1)\ K$	pK^2	K^2	$K\log_2 K$	...
MLP					
Train	$29W^3 + (20 + K)W^2 + (6 + K)$ $W + (63 + 5m)K - 42$	$29W^3 + (19 + K)W^2 + (2 + K)$ $W + (61 + 5m)K - 42$	...	$K\log_2 W$	$10K$
Test	$(5p+61)K$	$(5p+61)K$	...	...	$10K$

5.3. Results

The performance of the k-NN algorithm for AMC is investigated for different k values, which vary from 1 to 50 in 1-tap FFCs and 4-tap FSCs, respectively. Figures 8 and 9 show the performance of the correct classification percentage for the k-NN–based AMC in these 2 channel distributions. It is clearly seen that 66.43% of the classification performance in the FFC with 1-tap and 57.88% of the classification performance in the FSC with 4 taps to assign a correct MCS class label is obtained when k = 2. The major advantage of the small k value is the minimization in the processing time.

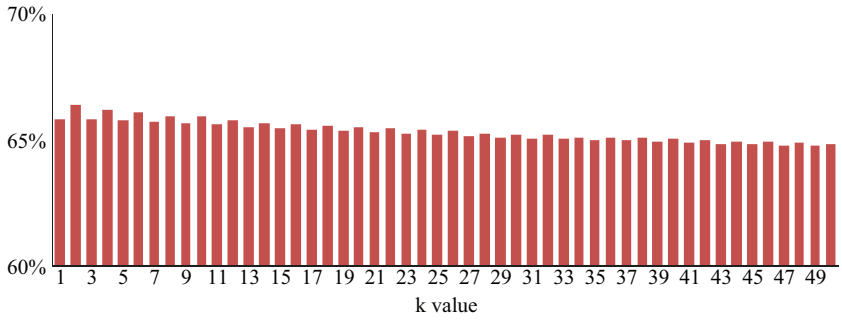

Figure 8. The correct classification percentages of the k-NN-based AMC for the different values of k in the FFCs with 1 tap.

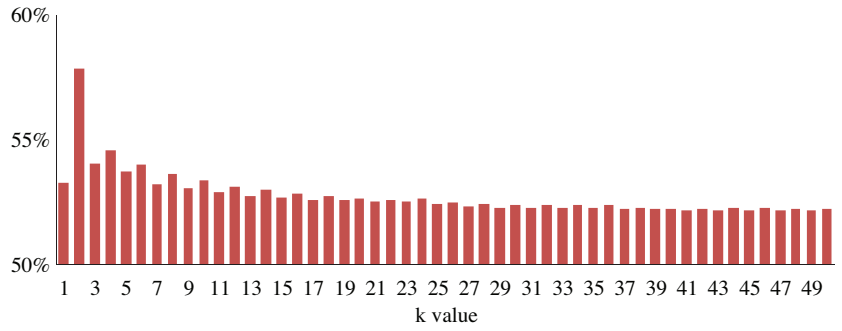

Figure 9. The correct classification percentages of the k-NN-based AMC for the different values of k in the FSCs with 4 taps.

Table 5 demonstrates the correct classification percentages of the MLP-based AMC for the different network configurations in the FFC with 1 tap and the FSC with 4 taps. To find the optimum network structure, the MLP networks each have the different hidden layers and neuron numbers depicted in Figure 6, run 500 times. The transfer function of the hidden layers is a tangent-sigmoid function, as given by Eq. (9). The output layer has a linear transfer function, as given by Eq. (10). The performance function of the network is chosen as the mean squared error. The maximum epoch number is set to 10,000. The learning rule selected is the LM backpropagation algorithm because of its convergence speed.

Table 5. The correct classification percentages of the MLP AMC for the different MLP models in the FFC and the FSC.

MLP model	FFC (1-tap)			FSC (4-tap)			Mean deviation
	Min	Max	Mean	Min	Max	Mean	
$S_1 = 2$	61.40%	64.33%	63.31%	60.92%	61.69%	61.44%	1.23%
$S_1 = 5$	61.38%	66.08%	64.01%	61.35%	61.72%	61.53%	1.69%
$S_1 = 10$	61.56%	66.79%	65.46%	61.36%	61.75%	61.56%	1.87%
$S_1 = 5$, $S_2 = 2$	63.67%	66.84%	65.61%	61.42%	61.78%	61.61%	1.18%
$\boldsymbol{S_1 = 5, S_2 = 5}$	**64.02%**	**66.89%**	**66.52%**	**61.39%**	**61.78%**	**61.61%**	**1.09%**
$S_1 = 5$, $S_2 = 2$, $S_3 = 5$	63.93%	66.87%	66.07%	61.13%	61.81%	61.62%	1.21%

In each run, networks are trained, and following that, they are tested. It is clearly seen that the best classification performance and the least deviation among the min, max, and mean of the 500 classification percentage values is achieved in the MLP network with 2 hidden layers, where each has $S_1 = 5$ and $S_2 = 5$

neurons. Consequently, we consider this MLP structure as an optimal network for the considered application and compare it with the k-NN (k = 2) algorithm in this paper.

Figure 10 shows the average PERs and spectral efficiencies as a function of the average SNR over all of the subcarriers and spatial streams from Eq. (5) in the FFCs with a single tap. The spectral efficiencies are obtained from Eq. (4). In Figure 10, the targeted AMC presents the case formulated by Eq. (2), not including the selection of the lower-class label indexed by a positive number. It is clearly seen that the k-NN AMC provides almost the same PER and spectral efficiency performance compared to the targeted AMC. The proposed MLP AMC performs with a better PER but slightly worse spectral efficiency performance (plots of the targeted AMC and k-NN AMC are overlapping). This is because the MLP guarantees a MCS with a lower data rate by way of the selection of the class label with a lower index number. Thus, the MLP approach provides higher communication robustness over the k-NN. In addition, our PER constraint (0.1% PER) is met after approximately a 2 dB average SNR.

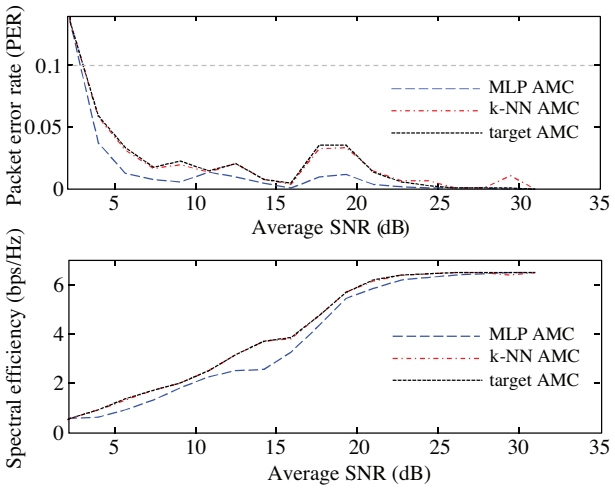

Figure 10. The PER and spectral efficiency for the MLP and k-NN AMC in the 1-tap FFC.

Figure 11 shows the PER results as a function of the average SNR when the k-NN AMC and MLP AMC are applied in the FSC with 4 taps. It is clearly seen that both classifiers meet the PER constraint when the average SNR is approximately more than 8.6 dB, and the MLP AMC outperforms the k-NN AMC. This is because the MLP ensures a MCS with a lower data rate through the selection of the lower index number, as mentioned before.

In Figure 12, the spectral efficiencies of both classifiers are plotted against the average SNR. As is clearly seen, the MLP AMC causes a barely decreasing spectral efficiency performance compared with the k-NN AMC.

Finally, we compare the percentages of the MCSs correctly adapted to the FFC with 1 tap and the FSC with 4 taps in the case of the MLP and the k-NN AMC in Figure 13. Obviously, the classification performance of the MLP-based AMC is better than that of the k-NN–based AMC in the FSC condition and nearly the same as the k-NN in the FFC. For our AMC application, we conclude that the selection of the MLP or k-NN classifier depends on the trade-off between the PER and the spectral efficiency, which depends on the end user requirements in both of the frequency channels.

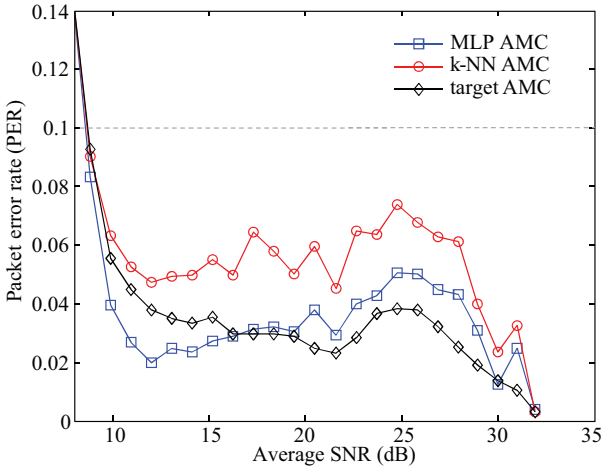

Figure 11. The PER as a function of the average SNR when the MLP AMC and k-NN AMC are applied in the 4-tap FSC.

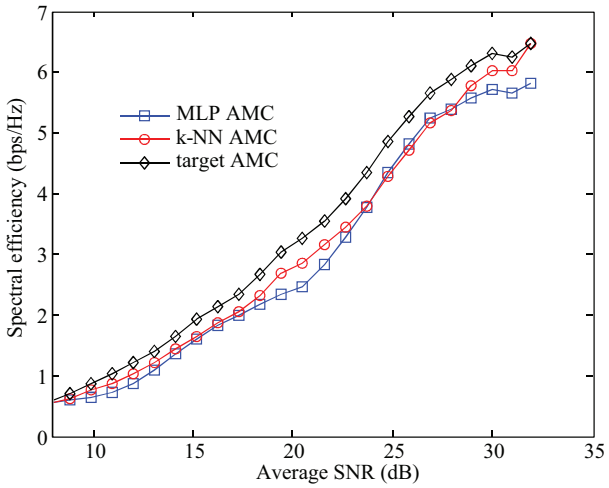

Figure 12. The spectral efficiency as a function of the average SNR when the k-NN AMC and MLP AMC are applied in the 4-tap FSC.

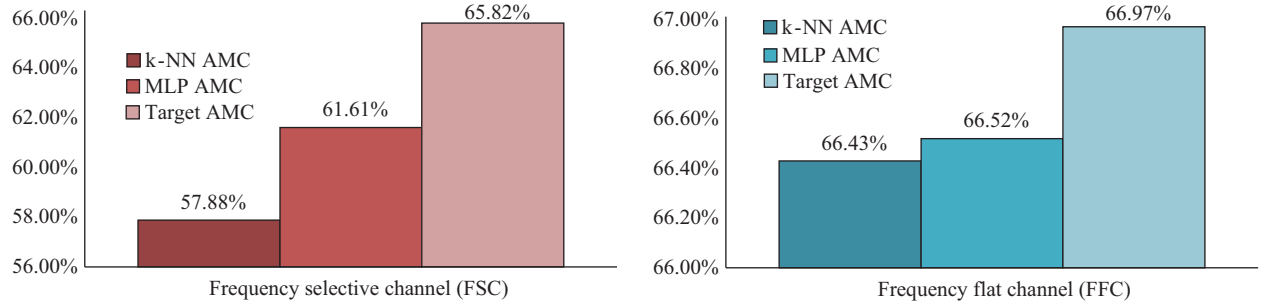

Figure 13. Correct classification percentages of the MLP AMC and k-NN AMC in the FFCs and the FSCs.

6. Conclusion

In this paper, we have proposed the MLP network for link adaptation based on AMC for the FFCs and the FSCs in a MIMO-OFDM system. We analyzed the performance of the PER and spectral efficiency of the MLP network, comparing it with the k-NN classifier. We also compared the percentages of the MCSs correctly adapted to the frequency channels in the case of the MLP and k-NN AMC. The simulation results demonstrated that the classification performance of the MLP-based AMC was better than that of the k-NN–based AMC in the FSC condition and was nearly the same as the k-NN in the FFC. The MLP AMC exhibited better PER performance compared to that of the k-NN AMC. The proposed MLP model selects a MCS with a lower data rate through the selection of the lower index number, and so it offers higher communication robustness. Consequently, the NN approach is applicable for the MCS selection in the MIMO-OFDM system for the frequency channels due to the fast mapping between the postprocessing SNR and the MCS index. Finally, we can finally conclude that the selection of the AMC classifiers depends on the trade-off between the PER and the spectral efficiency (data rate), depending on the user's requirements.

Acknowledgments

This work was supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under the 2214 PhD Research Fellowship Program. The authors would like to thank Robert W Heath Jr and Robert

C Daniels at the Department of Electrical and Computer Engineering at The University of Texas, Austin, for providing support.

References

[1] H. Yigit, A. Kavak, K. Kucuk, "Capacity improvement for TDD-MIMO systems via AR modeling based linear prediction", Wireless Personal Communications, Vol. 52, pp. 411–418, 2010.

[2] E. Telatar, "Capacity of multi-antenna Gaussian channels", European Transactions on Telecommunications, Vol. 10, pp. 585–595, 1999.

[3] S. Catreux, V. Erceg, D. Gesbert, R.W. Heath Jr, "Adaptive modulation and MIMO coding for broadband wireless data networks", IEEE Communications Magazine, Vol. 40, pp. 108–115, 2002.

[4] P.H. Tan, Y. Wu, S. Sun, "Link adaptation based on adaptive modulation and coding for multiple-antenna OFDM system", IEEE Journal on Selected Areas in Communications, Vol. 26, pp. 1599–1606, 2008.

[5] Y.S. Choi, S.A. Alamouti, "Pragmatic PHY abstraction technique for link adaptation and MIMO switching", IEEE Journal on Selected Areas in Communications, Vol. 26, pp. 960–971, 2008.

[6] H.T. Nguyen, J.B. Andersen, G.F. Pedersen, "On the performance of link adaptation techniques in MIMO systems", Wireless Personal Communications, Vol. 42, pp. 543–561, 2007.

[7] T. Javornik, S. Plevel, G. Kandus, "A recursive link adaptation algorithm for MIMO systems", Proceedings of the 12th IEEE Mediterranean Electrotechnical Conference, pp. 429–432, 2004.

[8] S. Simoens, S. Rouquette-Léveil, P. Sartori, Y. Blankenship, B. Classon, "Error prediction for adaptive modulation and coding in multiple-antenna OFDM systems", Signal Processing, Vol. 86, pp. 1911–1919, 2006.

[9] M. Lampe, T. Giebel, H. Rohling, W. Zirwas, "PER-prediction for PHY mode selection in OFDM communication systems", Proceedings of the Global Telecommunications Conference, Vol. 1, pp. 25–29, 2003.

[10] M. Sandell, "Link adaptation for MIMO systems using reliability values", Proceedings of the IEEE Wireless Communications and Networking Conference, Vol. 3, pp. 1608–1613, 2006.

[11] R.C. Daniels, C.M. Caramanis, R.W. Heath Jr, "Adaptation in convolutionally coded MIMO-OFDM wireless systems through supervised learning and SNR ordering", IEEE Transactions on Vehicular Technology, Vol. 59, pp. 114–126, 2010.

[12] S. Haykin, Neural Networks and Learning Machines, 3rd ed., New York, Prentice Hall, 2009.

[13] E. Alpaydin, Introduction to Machine Learning, Massachusetts, Cambridge, MIT Press, 2004.

[14] X. Wu, V. Kumar, J.R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. McLachlan, A. Ng, B. Liu, P.S. Yu, Z.H. Zhou, M. Steinbach, D.J. Hand, D. Steinberg, "Top 10 algorithms in data mining", Springer Journal on Knowledge and Information Systems, Vol. 14, pp. 1–37, 2008.

[15] Kavak, H. Yigit, H.M. Ertunc, "Using adaline neural network for performance improvement of smart antennas in TDD wireless communications", IEEE Transactions on Neural Network, Vol. 16, pp. 1616–1625, 2005.

[16] R.C. Daniels, C.M. Caramanis, R.W. Heath Jr, "A supervised learning approach to adaptation in practical MIMO-OFDM wireless systems", Proceedings of the Global Telecommunications Conference, pp. 1–5, 2008.

[17] H. Yigit, A. Kavak, "Adaptation using neural network in frequency selective MIMO-OFDM systems", Proceedings of the 5th IEEE International Symposium on Wireless Pervasive Computing, pp. 390–394, 2010.

[18] R. Daniels, R.W. Heath Jr, "Online adaptive modulation and coding with support vector machines", Proceedings of the IEEE European Wireless Conference, pp. 718–724, 2010.

[19] S. Yun, C. Caramanis, "Multiclass support vector machines for adaptation in MIMO-OFDM wireless systems", Proceedings of the 47th Annual Allerton Conference on Communication, Control, and Computing, 2009.

[20] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, Cambridge, Cambridge University Press, 2000.

[21] IEEE 802.11n Working Group, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications - Draft 5.0: Enhancements for Higher Throughput, Part 11, Standard Edition, 2007.