STRATEGIC OPTIMIZATION OF CONVOLUTIONAL NEURAL NETWORKS FOR HYPERSPECTRAL LAND COVER CLASSIFICATION

C. Buehler [1], F. Schenkel [2], W. Gross [2,] G. Schaab[1], W. Middelmann [2,] *

[1] Karlsruhe University of Applied Sciences Karlsruhe, Faculty for Information Management and Media, Germany – (buca1013, gertrud.schaab)@hs-karlsruhe.de
[2] Fraunhofer IOSB, Ettlingen, Germany – info-ettlingen@iosb.fraunhofer.de

KEY WORDS: Hyperspectral Imagery, CNN, Transfer Learning, Classification, Spectral Feature Extraction, CNN Architecture Optimization

ABSTRACT:
Hyperspectral data recorded by future earth observation satellites will have up to hundreds of narrow bands that cover a wide range of the electromagnetic spectrum. The spatial resolution (around 30 meters) of such data, however, can impede the integration of the spatial domain for a classification due to spectrally mixed pixels and blurred edges in the data. Hence, the ability of performing a meaningful classification only relying on spectral information is important. In this study, a model for the spectral classification of hyperspectral data is derived by strategically optimizing a convolutional neural network (1D-CNN). The model is pre-trained and optimized on imagery of different nuts, beans, peas and dried fruits recorded with the Cubert ButterflEye X2 sensor. Subsequently, airborne hyperspectral datasets (Greding, Indian Pines and Pavia University) are used to evaluate the CNN's capability of transfer learning. For that, the datasets are classified with the pre-trained weights and, for comparison, with the same model architecture but trained from scratch with random weights. The results show substantial differences in classification accuracies (from 71.8% to 99.8% overall accuracy) throughout the used datasets, mainly caused by variations in the number of training samples, the spectral separability of the classes as well as the existence of mixed pixels for one dataset. For the dataset that is classified least accurately, the greatest improvement with pre-training is achieved (difference of 3.3% in overall accuracy compared to the non-pre-trained model). For the dataset that is classified with the highest accuracy, no significant transfer learning was observed.

1. INTRODUCTION

In recent years, hyperspectral imaging has become an important research field (Bioucas-Dias et al., 2013). Dozens to hundreds of narrow bands that cover a wide range of the electromagnetic spectrum offer new possibilities in a large field of applications, such as land cover mapping or material identification (Petropoulos et al., 2012; Rast and Painter, 2019). Currently, the acquisition of hyperspectral data from space is hampered since no imaging spectrometer exists yet that records publicly available data. In the near future, however, the launch of a number of sensors is planned (Pu, 2017; Rast and Painter, 2019). Hyperspectral data from space usually comes with a spatial resolution of around 30 meters (Guanter et al., 2015), which may lead to mixed pixels and blurred edges in the data. Hence, it can be assumed that the usage of the spatial domain for a classification is exacerbated for such data. Thus the ability to perform a meaningful classification purely relying on spectral information becomes important.

Among others, deep learning based classification methods have proven to perform well when handling high-dimensional data such as hyperspectral imagery. Particularly *Convolutional Neural Networks* (CNNs) achieve superior performance in classifying images compared to traditional methods, such as *Support Vector Machines* (Gao et al., 2018). Moreover, CNNs are capable of transfer learning from a source task to a related target task (Yang et al., 2017).

However, the design of an appropriate CNN is difficult due to a large number of tunable parameters. Moreover, the optimal CNN is highly dependent on the task to be solved (Baker et al., 2016). A number of studies have been conducted that classify hyperspectral data with deep learning methods. Their approaches vary in terms of models applied, domains used, or whether training is performed in a supervised or unsupervised manner. The highest classification accuracies are achieved among the studies that performed supervised classification in the spectral-spatial domain (Makantasis et al., 2015; Santara et al., 2017; Tao et al., 2015). Less frequent studies that classified hyperspectral data in the spectral domain achieved comparably lower accuracies (Hu et al., 2015).
In this paper, we present our research for extracting spectral features from hyperspectral imagery with 1D-CNNs, i.e. a CNN that takes a one-dimensional spectrum as input. For that, a large amount of hyperspectral data (about 350,000 samples) of various object types (nuts, beans, peas and dried fruits) is acquired using the Cubert ButterflEye X2 imaging spectrometer. This dataset, subsequently termed Cubert dataset, is used to strategically optimize a 1D-CNN architecture. Detailed analysis on the influence of CNN hyper-parameter tuning is conducted to examine the impacts on the overall accuracy (OA) of the classification result. The CNN that is pre-trained on the Cubert data is then used to evaluate the model's capability of transfer learning on three airborne hyperspectral benchmark datasets. For that, the spectra of each dataset are once classified with the pre-trained model and once with the same model architecture, but trained from scratch.

This paper is organized as follows: Section 2 describes the used datasets and how they are pre-processed. Moreover, the conducted strategy for CNN optimization and transfer learning is outlined. In Section 3, results, the order of CNN optimization by hyper-parameter tuning is shown and the best-performing model is presented. In addition, the classification results for the benchmark data are listed with a comparison for the pre-trained

* Corresponding author

versus the non-pre-trained model. Section 4 discusses the results delineated in Section 3 with respect to the general findings that are made during CNN optimization, the evaluation of the classification results and the benefits that are gained by pre-training. Section 5 concludes the study by summarizing the results and providing an outlook for possible future research.

2. METHODS

2.1 Datasets

The Cubert dataset, which is used for optimizing the CNN consists of 25 bands in a spectral range that covers the visible to the near infrared spectrum (475 – 875 nm) with each band having a width of 20 nm[1]. 15 different object types (classes) are recorded and shown in Table 1 together with an overview of the available samples.

Class	Number of samples	Class	Number of samples
Almond	26,785	Green Split Pea	11,182
Blanched Almond	49,228	Hazelnut	17,908
Blue Raisin	13,158	Kidney Bean	38,776
Brazil Nut	26,332	Pinto Bean	23,360
Cashew Nut	26,547	Sultana	12,363
Common Jack Bean	24,076	Walnut	39,910
Cranberry	22,430	Yellow Split Pea	10,900
Goji Berry	7,421		

Table 1. Overview of the used object types for acquiring the Cubert dataset.

These object types are chosen as the available imaging spectrometer acquires terrestrial data and therefore, no data about land cover can be recorded. It is assumed that the extraction of features from the recorded object types is not a vastly different application for a CNN than extracting e.g. land cover features from airborne data, as the model should learn to interpret the same patterns in the data, such as gradients, minima or maxima. When selecting the objects, it is considered to also choose object types that are similar in color, e.g. blanched almonds and cashew nuts. However, the spectra recorded by the hyperspectral sensor should ideally not be similar, but this is not known in advance.

In total, 80 single images on six different image scenes are acquired. Figure 1 shows two sample image scenes. From band 12 to 25, the Cubert data is noisy for an unknown reason. This is considered later when evaluating the classification results.

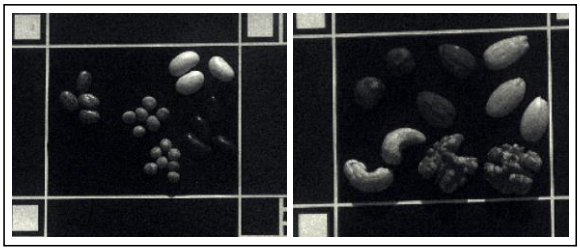

Figure 1. Exemplary image scenes of the Cubert dataset (band 1, mean wavelength 482.98 nm).

[1] According to the brochure: *Real-Time-Spectral Imaging: Introducing the ButterflEYE X2* (https://sphereoptics.de/wp-content/uploads/2015/01/S258_flyer.pdf, accessed April 29, 20)

In addition to the Cubert dataset, three airborne hyperspectral benchmark datasets, namely Greding, Indian Pines, and Pavia University, are used to evaluate the CNN's ability of transfer learning (Gross et al., 2019; Baumgardner et al., 2015; Plaza et al., 2006. Table 2 shows details on the all used datasets.

2.2 Data Pre-processing

The Cubert data comes with a randomly distributed spatial offset with maximum four pixels in the different bands. This may be caused by inaccuracies in the alignment of the sensor's grating. The offset is removed by shifting all bands by their relative offset to match the first band.

	Cubert	Greding	Indian Pines	Pavia University
Sensor	Cubert ButterflEye X2	asiaEagle II	AVIRIS	ROSIS-03
Spectral range (nm)	475 -875	390 - 990	400 - 2450	430 - 860
Number of bands	25	127	200	103
Spatial resolution (m)	approx. 0.0007	0.5	20	1.3
Dimension (pixels)	510 x 255	670 x 606	145 x 145	640 x 340
Number of classes	15	6	16	9
Number of samples	346,921	359,616	10,366	42,776

Table 2. Overview of the used datasets (Gross et al., 2019; Baumgardner et al., 2015; Plaza et al., 2006).

In addition, all pixels depicting one of the 15 object types are labelled with the corresponding class name. This resulted in 346,921 samples. From the 80 available images, 72 are used for training and validation, while the remaining images were used for testing the CNN. It is ensured that for both training and validation, data of all image scenes are used. In addition, the data is shuffled randomly for improving the statistical validity of the model (Wan et al., 2018).

The Cubert data is then processed using a min-max-normalization (Jayalakshmi and Santhakumaran, 2011). The benchmark datasets are already pre-processed (Gross et al., 2019; Baumgardner et al., 2015; Plaza et al. 2006). Hence, it is not necessary to conduct other pre-processing procedures than training / validation / test data splitting and the calculation of a normalized dataset with the same technique as used for the Cubert data.

2.3 CNN Architecture Optimization

The initial CNN is implemented similar to the model of Hu et al., (2015), since this model was designed for a related task, i.e. spectral classification of hyperspectral imagery. In addition, the CNN architecture of Hu et al., (2015) consists of an assessable amount of layers. The idea is that when starting with a basic CNN, the influence of changes in architecture can be monitored more precisely.

The CNN of Hu et al., (2015) consists of five layers in total. The first layer is the input layer with the dimensions n_1 x 1 since each pixel is regarded as a vector of values, while n_1 corresponds to the number of bands. The subsequent convolutional layer C_2 filters the data with 20 kernels and a *hyperbolic tangent function* (tanh) is applied for activation. The second hidden layer, pooling layer P_3, applies a *max pooling* function for downsampling. The next layer is a fully-connected layer termed F_4, also applying the tanh activation function. Last layer of the initial CNN is an output layer, which is also fully-connected and classification is performed using a soft-max function.

After setting up the initial CNN, its architecture is optimized iteratively by applying the following changes to the model:

- Adapting hyper-parameters of convolutional / pooling / fully-connected layers, e.g. *stride*, *zero-padding*, *receptive field*, *pooling* / *activation functions* (Khan et al., 2018)
- Extending / shrinking of model size, e.g. adding / removing layers, adding / removing connections (Khan et al., 2018)
- Changing training parameters, e.g. *learning rate*, *loss functions*, *batch size*, *epochs*, *weight initialization*, training / validation data distribution (Khan et al., 2018)
- Regularization methods, e.g. *dropout*, *batch normalization* (Ioffe and Szegedy, 2015; Srivastava et al., 2014)

After each adaptation, the model performs the classification of the Cubert data. Then, it is evaluated whether the classification's overall accuracy (OA), i.e. the percentage of correctly predicted samples, is improved compared to the previous architecture. If yes, the adaptation is adopted. Optimizing the CNN architecture is stopped after OA becomes stable for a number of optimization iterations. The best-performing model is selected with respect to OA and over-fitting.

2.4 Transfer Learning

The CNN that is pre-trained on the Cubert dataset is next used to classify the three hyperspectral benchmark datasets (section 2.1). For evaluating the pre-trained CNN's ability of transfer learning, the same CNN architecture is built than finally used for the Cubert data. The models only differ in their input (number of bands) and output (number and type of classes) dimensions as this is different for each dataset.

In a CNN, the convolutional layers perform the feature extraction, while fully-connected layers are used for classification (Hu et al., 2015). For the benchmark datasets, different classes than for the Cubert data should be distinguished, e.g. trees vs. walnuts. Hence, the fully-connected layers that are trained to classify the Cubert data are not assumed to know how to classify the benchmark data. For the feature extraction performed by the convolutional layers, however, similar tasks should be solved for both models, i.e. distinguishing classes based on specific spectral features. Thus, it is anticipated that the knowledge the classifier gained during pre-training may be helpful for detecting features in the spectral signatures of the benchmark data. Hence, transfer learning is only conducted for the feature extraction, i.e. convolutional layers, while the class assignment conducted by fully-connected layers is learned from scratch.

Each dataset is classified with and without pre-training the convolutional layer to directly examine its influence. This is comparable with the approach of Yang et al. (2017). The weights of the CNN are directly applied to datasets with different spectral ranges to evaluate the capability of universal feature representation using only data from a single hyperspectral sensor.

3. RESULTS

3.1 Model Optimization

Starting with a CNN similar to the one of Hu et al. (2015), numerous variations in CNN architecture are tested. 28 of 53 conducted runs that show the most important adaptations concerning the improvement of OA are visualized in Figure 2.

Initially, the best-performing pre-processing method for the given data is selected. Thereafter, the activation function and the weight initialization are tuned together with adaptations on the hyper-parameters of the convolutional layer. The learning rate is lowered while the number of epochs is increased. As the model tends to over-fit, dropout is integrated then. Next step is to work on the model size by adding and removing convolutional and fully-connected layers. Subsequently, training parameters are optimized again and batch normalization is introduced for reducing over-fitting. The classification's accuracy (OA) becomes stable at around 77% eventually. Hence, it is presumed that the improvability of the OA is limited to a certain extent and CNN modification is terminated.

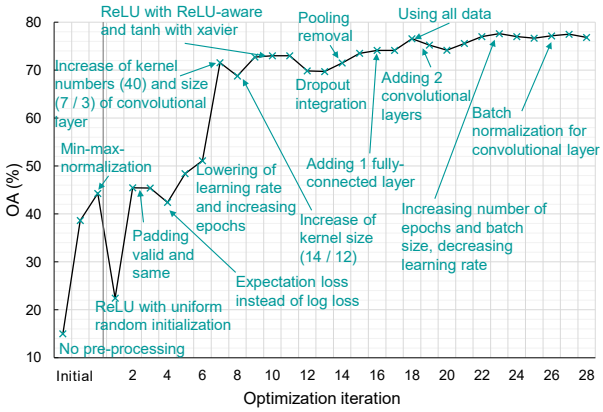

Figure 2. Most important adaptations concerning the improvement of model accuracy (OA of the test set).

3.2 Identification of the Best-Performing CNN Architecture

For selecting the model that proves best for the Cubert dataset, the three models that classified the data with the highest OA are compared with respect to OA, kappa coefficient, f1-scores and over-fitting. An Attempt is made to choose the model that gives a tradeoff of all metrics.

Figure 3 shows the CNN that proves best for the Cubert data. It consists of an input layer *I* that takes a vector of 25x1 values as input. The following convolutional layer *C* filters the data with 40 kernels of size 13x1 and is followed by a batch normalization layer *BN* for reducing its internal covariance shift (Ioffe and Szegedy, 2015). Subsequently, the output is flattened to use it as input for the two following fully-connected layers FC_1 and FC_2. Dropout (D_1, D_2) with a rate of 0.4 is applied directly after the first two fully-connected layers in order to reduce over-fitting. The last layer FC_3 performs the actual classification by applying a *soft-max* function on its input. For all other layer activations, *rectified linear unit* (ReLU) activation function is applied and ReLU-aware weight initialization is chosen. The learning rate is set to 0.0001 and *log loss* function is applied. The model is trained in 280 epochs and with a batch size of 75. Using the described model, the Cubert test dataset is classified with an OA of 77.1%, a kappa coefficient of 74.6%.

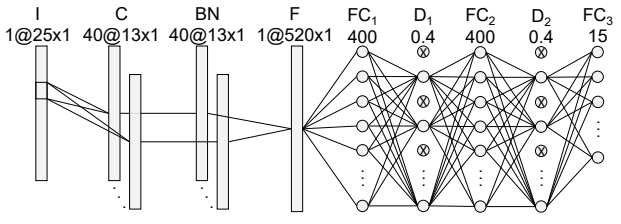

Figure 3. Architecture of the proposed CNN for Cubert data classification.

The first other model that performs similar and is a candidate for the best-performing CNN has an almost equal architecture, except having 300 units for the fully-connected layers and a dropout rate of 0.2. This model tends to over-fit and has a comparable OA to the selected model. In comparison, the second alternative model that performs similar is deeper. It consists of four convolutional layers with 80 kernels and six fully-connected layers also regulated by batch normalization respectively dropout. This model produces a comparable accuracy and little over-fitting.

For gaining insight into errors affecting individual classes, f1-scores for each class of the Cubert dataset are calculated (Table 3). The average f1-score is 76.3. There is a huge spread in values for the f1-score ranging from 51.8 to 91.6 throughout the different classes, which reveals that not all classes are predicted equally well by the model. For instance, 'Hazelnut' comes with the lowest f1-score, while 'Kidney Bean' and 'Common Jack Bean' have significantly higher scores than all other classes.

Class	F1-score	Class	F1-Score
Almond	71.4	Green Split Pea	83.4
Blanched Almond	82.5	Hazelnut	51.8
Blue Raisin	80.1	Kidney Bean	91.1
Brazil Nut	69.3	Pinto Bean	70.4
Cashew Nut	67.5	Sultana	62.2
Common Jack Bean	91.6	Walnut	74.2
Cranberry	82.9	Yellow Split Pea	82.2
Goji Berry	84.5		

Table 3. F1-scores for the classification results of the Cubert dataset with the CNN of Figure 3.

3.3 Transfer Learning for Benchmark Data Classification

A direct comparison between the classification performed by the pre-trained model and the same CNN trained from scratch is conducted for each of the three benchmark datasets. For the Greding dataset, the model is trained on 71% of the samples, 21% are used for validation and 8% for testing. The Indian Pines data is trained on 66% of the samples, on 17% validation is performed and 17% are used for testing. In contrast, from the Pavia University data, 72% are used for training, 22% for validation and 6% for testing the model. Table 4 shows the resulting classification metrics that vary substantially between the three benchmark datasets.

The classification results of the Greding dataset are similar with and without pre-training and achieve an OA of 99.7% and 99.8%, respectively. In comparison, the Indian Pines dataset is classified with lower accuracies of 68.5% and 71.8%. For this dataset, a substantial improvement in all metrics is achieved when using the pre-trained weights. In contrast, the samples of the Pavia University dataset are predicted with an OA of 91.5% to 92.8%. This is a considerably higher OA than achieved for the Indian Pines dataset. However, the accuracy improvement with pre-training is not as high as for the Indian Pines dataset (3.3% in OA compared to 1.3%).

	Greding		Indian Pines		Pavia University	
	NPT	PT	NPT	PT	NPT	PT
Overall accuracy	99.8	99.7	68.5	71.8	91.5	92.8
Kappa coefficient	99.7	99.7	63.8	67.4	88.8	90.5
Average f1-score	100.0	100.0	68.0	70.0	92.0	93.0

Table 4. Classification metrics for the three airborne hyperspectral datasets. NPT refers to the *n*on-*p*re-*t*rained classification model, while PT denotes the *p*re-*t*rained CNN. The metrics are computed based on the independent test set.

4. DISCUSSION

4.1 Findings during Model Optimization

During the optimization procedure of the CNN for the Cubert dataset, a number of findings are made. These are listed and discussed subsequently. For viewing the order of parameter optimization, refer to Figure 2.

4.1.1 Pre-processing: *Min-max-normalization* proves to be the most suitable pre-processing technique since significant accuracy improvement compared to the other techniques is achieved. This supports the finding of Pal and Sudeep (2016), saying that no pre-processing results in lower accuracies.

4.1.2 Activation Function and Weight Initialization: *ReLU* activation function combined with the *ReLU aware scaled initialization* produces similar results than combining *tanh* activation function with *Xavier initialization* (Glorot and Bengio, 2010). This is in accordance with the findings of Kumar (2017).

4.1.3 Convolutional Layer: More kernels for convolutional layers can improve model accuracy. However, the more kernels are used, the slower the model fits. 40 kernels tend to be the best trade-off between accuracy and training time. A kernel size between 3 and 7 produces the best results concerning OA.

4.1.4 Pooling Layer: Downsampling is not needed for classifying the Cubert dataset as the removal of pooling layers leads to an increase in OA of about 2%. As the input samples of the Cubert data are not of high dimensionality (25 x 1 bands), it is anticipated that for such data, dimensionality reduction with pooling layers can eliminate features that are important for classification, which could lead to lower accuracies as observed in this study.

4.1.5 Fully-Connected Layer: The optimal number of units for the Cubert dataset are 400 with two consecutive fully-connected layers since the best model performance is achieved with these parameter settings.

4.1.6 Training Parameters: The smaller the *learning rate*, the longer it takes until a CNN converges. Hence, more epochs are necessary (Khan et al., 2018). A learning rate of 0.0001 and 280 epochs tends to be appropriate for the Cubert dataset. *Expectation loss* shows inferior performance than *log loss* function (difference in OA of about 3%). A maximum *batch size* of 120 is tested (not shown in Figure 2), but 75 performs best, also compared to the initial setting of 25.

4.1.7 Model Size / Layer Order: One or two convolutional layers are sufficient for successful feature extraction. More convolutional layers do not lead to better results. Besides, training time is increased substantially when having more than two convolutional layers.

A deeper model with four convolutional layers and six fully-connected layers achieves an OA comparable with more simple architectures, but training time is increased significantly. Hence, the architecture of Figure 3 is considered appropriate for the given data.

4.1.8 Regularization: Introducing *batch normalization* after convolutional layers and *dropout* (rate 0.4) after fully-connected layers is helpful for reducing over-fitting.

4.2 Evaluation of Classification Results

4.2.1 Cubert Data: Considering the OA of 77.1% together with the varying f1-scores and a confusion matrix, the classification results are evaluated by also observing spectra of each class (Table 3). By that, it is detected that mainly similar spectral features and also the noisy data must be responsible for most misclassifications (Figure 4).

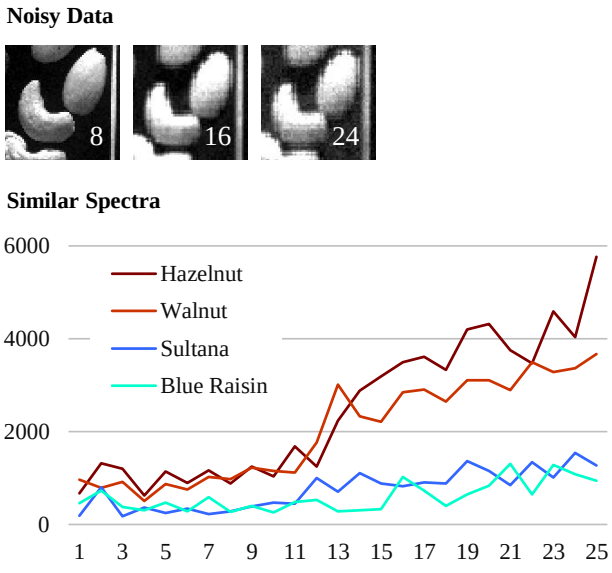

Figure 4. Noisy data and similar spectra as possible reasons leading to misclassification of the Cubert samples. The images showing the noisy data are labelled with their band number. The x-axis of the spectra graph refers to the band numbers and the y-axis to the corresponding reflectance values of a pixel.

For instance, Hazelnut is frequently confused with Walnut (16.3%), while Sultana is often misclassified as Blue Raisin (18.8%). With respect to Figure 4 it can be stated that these examples show similar spectral features that probably causes misclassifications. In case of the classes Hazelnut and Walnut it can be said that the spectra are similar mainly in the first 12 bands. Noise starts from band 12, which may also cause misclassifications. Moreover, it is possible that classification accuracy could be improved when using data from a sensor with a higher spectral resolution.

It can be summarized that several classes mix up with others, which also causes a lowering of classification accuracy. It is assumed that this mostly comes from similar spectral features as well as the noise present from band 12 of the Cubert data. Therefore, it is anticipated that when pre-processing the data with respect to noise removal and when having classes with no spectral feature overlapping, classification accuracy would rise substantially.

4.2.2 Benchmark Data: For the Greding dataset, it is assumed that the nearly perfect predictions for this dataset come from unique spectral features for all six classes of the Greding dataset. Other factors that may affect the classifiers ability to reliably distinguish the classes are the comparable low amount of classes so that inter-class spectral mixture can be avoided as well as the relatively high number of available samples as compared to other benchmark datasets. Compared to other, related studies, the CNN of this study performed differently for the benchmark data. For the Greding dataset, no study was conducted yet that used a CNN. A study accomplished by Gross et al. (2019), however, achieved lower accuracies (difference around 1 to 3%) using a spectral angle mapper (SAM) and support vector machines (SVM).

The Indian Pines dataset consists of much less labelled samples than the Greding data and a higher number of classes. The spatial resolution could cause the 'mixed-pixel-problem' and the ground truth data is erroneous[2]. Hence, the Indian Pines data is a complex dataset that comes with a number of influences that can affect a classifier's performance. This could explain the vast difference in classification accuracy compared to the other datasets. When comparing the classification results for the Indian Pines dataset to those of related studies, it becomes apparent that the classifiers of the other studies performed better. For instance, the model proposed by Hu et al. (2015) predicted 90.2% of the samples correctly by using a CNN for the spectral domain. However, Hu et al. (2015) discarded eight classes that have less than 200 samples, which explains the significantly higher classification accuracy achieved in their study. The classification accuracies of Makantasis et al. (2015) are also higher (98.9%) than what is achieved in this study. However, Makantasis et al. (2015) did not only use the spectral information of the data, but also included spatial information of adjacent pixels.

The quality of the predictions the classifier made for the Pavia University lies with 92.8% between those of the other two datasets. The number of samples and classes also lies between the other datasets, which underlines the relationship of a classification's accuracy and the number of training samples respectively the number of classes and their spectral uniqueness. The spectra of the Pavia University dataset are mostly unique, but 9.6% - 16.5% of the samples of Asphalt, Bitumen, Gravel and Bare Soil are misclassified. Hu et al. (2015) achieved an OA of

[2] According to an open discussion at the IEEE International Geoscience and Remote Sensing Symposium (IGARSS) 2017

92.6% when classifying the Pavia University dataset in the spectral domain, which is similar to the results of this study. Makantasis et al. (2015) achieved 99.6% by considering not only the spectral, but also the spatial information of the Pavia University dataset.

The vast difference in classification accuracies when using a spectral compared to a spectral-spatial classification algorithm underlines the importance of conducting more research on performing a meaningful classification of hyperspectral data based on spectral information.

4.3 Benefits gained by Transfer Learning

For this study, the CNN that is trained on the large amount of Cubert data is used for transfer learning to classify three different hyperspectral benchmark datasets: Greding, Indian Pines and Pavia University. The benefits that are achieved by pre-training are assessed by comparing the difference in classification accuracies (OA) when using the model with pre-trained and random weights for the convolutional layer.

Classification results for the Greding data are similar with and without pre-training (Table 4). However, as the predictions are performed with very high accuracy already when not pre-training the model, almost no further improvement can be expected. Hence, no point can be made about the classifier's ability of transfer learning for this dataset.

As already stated in Section 4.2.2, the Indian Pines dataset is complex and comes with a number of influences that can affect a classifier's performance. However, by applying pre-training, a substantial improvement of classification accuracy is achieved.

Similar to Indian Pines, also the Pavia University dataset is classified with a higher accuracy when using the pre-trained convolutional layer for feature extraction. However, the accuracy improvement with pre-training is not as high as for the Indian Pines dataset. Additionally, the general performance of the classifier is higher for the Pavia University data, as a vastly higher OA was achieved.

Figure 5 compares the achieved OA with the amount of improvement achieved by pre-training for the three benchmark datasets. Based on this comparison, it can be stated that there is a direct relationship between the achievable accuracy of a classification and the improvement that can be gained by pre-training, i.e. that pre-training is beneficial particularly for datasets that are difficult to classify.

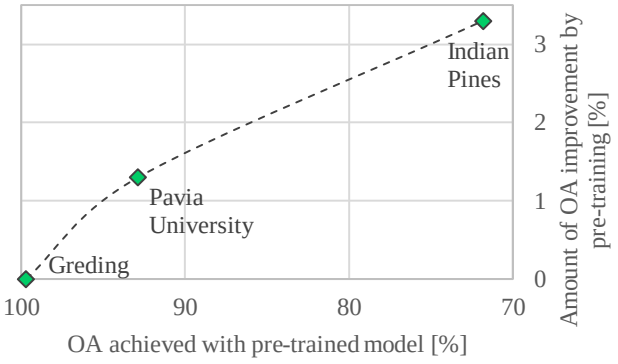

Figure 5. Comparison of the OA achieved for the classification of the three benchmark datasets and the OA improvement achieved by using the pre-trained model.

5. CONCLUSION

In this study, fundamental research in terms of spectral feature extraction from hyperspectral data in combination with CNN model optimization is conducted based on a transfer learning approach.

The experiments show that it is possible to distinguish visually similar objects using a CNN for spectral feature extraction. However, it is observed that the distinguishability is highly dependent on the uniqueness of the spectral features of a class since a high variance in accuracies for individual classes is noticed.

Moreover, this study has shown that CNN's are capable of transfer learning, as the non-pre-trained model produces less accurate results than the same model with pre-training for two of three datasets. Further, the results have shown that the poorer the classification accuracy for a dataset is, e.g. due to similar spectral features of the classes, a low amount of labelled data or the 'mixed-pixel problem', the greater is the benefit that can be gained by transfer learning. Since it is presumed that classifications suffer from a lower spatial resolution due to the 'mixed-pixel problem' and the existence of blurred edges, it can be stated that transfer learning is especially beneficial for such data, i.e. the target data of this research (spaceborne data acquired by future hyperspectral earth observation satellites).

For future research, it may be useful to conduct experiments with other classes for pre-training. When acquiring data in a terrestrial setting, more diverse object types could be selected by adding objects like metal sheets, asphalt, leaves or soil. By that, more diverse spectra would be learned by the classifier, which could make transfer learning more effective. Additionally, when using another imaging spectrometer that acquires data with a higher spectral resolution, i.e. having more bands covering a broader range of the electromagnetic spectrum, experiments could be conducted with respect to an improvement of the spectral separability and transfer learning. Moreover, for considering the different spectral ranges throughout the datasets, an appropriate model for each benchmark dataset could be designed and pre-trained with the Cubert data.

REFERENCES

Baker, B., Gupta, O., Naik, N., and Raskar, R. (2016). Designing neural network architectures using reinforcement learning. *Computing Research Repository*, *abs/1611*.02167.

Baumgardner, M. F., Biehl, L. L., and Landgrebe, D. A. (2015). 220 band AVIRIS hyperspectral image data set: Acquired June 12, 1992 on Indian Pine Test Site 3. doi: doi:10.4231/R7RX991C.

Bioucas-Dias, J. M., Plaza, A., Camps-Valls, G., Scheunders, P., Nasrabadi, N., and Chanussot, J. (2013). Hyperspectral remote sensing data analysis and future challenges. *IEEE Geoscience and Remote Sensing Magazine, 1*(2), 6-36. doi: 10.1109/MGRS.2013.2244672.

Gao, Q., Lim, S., and Jia, X. (2018). Hyperspectral image classification using convolutional neural networks and multiple feature learning. *Remote Sensing, 10*(2), 299. doi: 10.3390/rs10020299.

Glorot, X., & Bengio, Y. (2010, March). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 249-256.

Gross, W., Tuia, D., Soergel, U., and Middelmann, W. (2019). Nonlinear feature normalization for hyperspectral domain adaptation and mitigation of nonlinear effects. *IEEE Transactions on Geoscience and Remote Sensing*, *57*(8), 5975-5990. doi:10.1109/TGRS.2019.2903719.

Guanter, L., Kaufmann, H., Segl, K., Foerster, S., Rogass, C., Chabrillat, S., Kuester, T., Hollstein, A., Rossner, G., Chlebek, C., et al. (2015). The EnMAP spaceborne imaging spectroscopy mission for earth observation. *Remote Sensing*, *7*(7), 8830-8857. doi:10.3390/rs70708830.

Hu, W., Huang, Y., Li, W., Zhang, F., and Li, H. (2015). Deep convolutional neural networks for hyperspectral image classification. *Journal of Sensors*, 1-12. doi: 10.1155/2015/258619.

Ioffe, S., and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *32nd International Conference on Machine Learning, Lille*, 448-456.

Jayalakshmi, T. and Santhakumaran, A. (2011). Statistical normalization and back propagation for classification. *International Journal of Computer Theory and Engineering*, *3*(1), 1793-8201. doi:10.7763/IJCTE.2011.V3.288.

Khan, S., Rahmani, H., Shah, S. A. A., and Bennamoun, M. (2018). A guide to convolutional neural networks for computer vision. *Synthesis Lectures on Computer Vision*, *8*(1), 1-207.

Kumar, S. K. (2017). On weight initialization in deep neural networks. *Computing Research Repository*, abs/1704.08863.

Makantasis, K., Karantzalos, K., Doulamis, A., & Doulamis, N. (2015). Deep supervised learning for hyperspectral data classification through convolutional neural networks. In *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 4959-4962.

Pal, K. K. and Sudeep, K. (2016). Preprocessing for image classification by convolutional neural networks. In *IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore*, 1778-1781.

Petropoulos, G. P., Arvanitis, K., & Sigrimis, N. (2012). Hyperion hyperspectral imagery analysis combined with machine learning classifiers for land use/cover mapping. *Expert Systems with Applications*, *39*(3), 3800-3809. doi: 10.1016/j.eswa.2011.09.083.

Plaza, A., Benediktsson, J. A., Boardman, J., Brazile, J., Bruzzone, L., Camps-Valls, G., et al. (2006). Advanced processing of hyperspectral images. In *2006 IEEE International Symposium on Geoscience and Remote Sensing*, 1974-1978. doi: 10.1109 / IGARSS.2006.511.

Pu, R. (2017). Hyperspectral remote sensing: Fundamentals and practices. *CRC Press, Boca Raton*.

Rast, M. and Painter, T. H. (2019). Earth observation imaging spectroscopy for terrestrial systems: An overview of its history, techniques, and applications of its missions. *Surveys in Geophysics*, 1-29. doi:10.1007/s10712-019-09517-z.

Santara, A., Mani, K., Hatwar, P., Singh, A., Garg, A., Padia, K., and Mitra, P. (2017). Bass Net: Band-adaptive spectral-spatial feature learning neural network for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, *55*(9), 5293-5301. doi:10.1109/TGRS.2017.2705073.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929-1958.

Tao, C., Pan, H., Li, Y., and Zou, Z. (2015). Unsupervised spectral-spatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification. *IEEE Geoscience and Remote Sensing Letters*, *12*(12), 2438-2442. doi: 10.1109/LGRS.2015.2482520.

Wan, K., Tuninetti, D., Ji, M., and Piantanida, P. (2018). Fundamental limits of distributed data shuffling. In 2018 *56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 662-669. doi: 10.1109/ALLERTON.2018.8635882.

Yang, J., Zhao, Y.-Q., and Chan, J. C.-W. (2017). Learning and transferring deep joint spectral-spatial features for hyperspectral classification. *IEEE Transactions on Geoscience and Remote Sensing*, *55*(8), 4729-4742. doi: 10.1109/TGRS.2017.2698503.