Wavelength-aware 2D Convolutions for Hyperspectral Imaging

Leon Amadeus Varga, Martin Messmer, Nuri Benbarka, Andreas Zell
Cognitive Systems Group
University of Tuebingen
Tuebingen, Germany
leon.varga@uni-tuebingen.de, martin.messmer@uni-tuebingen.de,
nuri.benbarka@uni-tuebingen.de, andreas.zell@uni-tuebingen.de

Abstract

Deep Learning could drastically boost the classification accuracy for Hyperspectral Imaging (HSI). Still, the training on the mostly small hyperspectral data sets is not trivial. Two key challenges are the large channel dimension of the recordings and the incompatibility between cameras of different manufacturers.

By introducing a suitable model bias and continuously defining the channel dimension, we propose a 2D convolution optimized for these challenges of Hyperspectral Imaging. We evaluate the method based on two different hyperspectral applications (inline inspection and remote sensing). Besides the shown superiority of the model, the modification adds additional explanatory power.

In addition, the model learns the necessary camera filters in a data-driven manner. Based on these camera filters, an optimal camera can be designed.

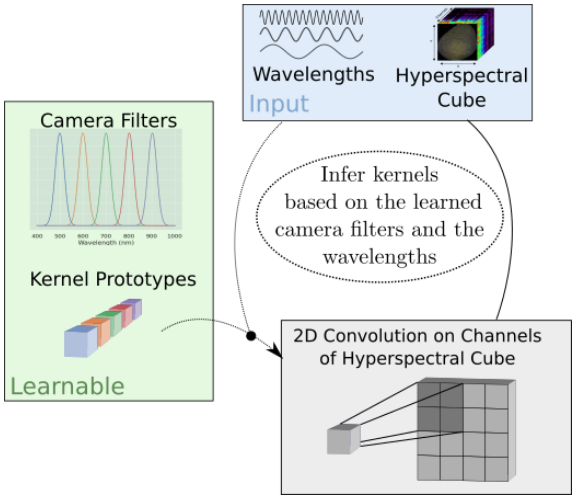


Figure 1: Hyperspectral Visual Embedding Convolution (HyveConv) at a glance. Further details in section 3.2.

1. Introduction

Image classification is one of the main tasks in computer vision [8]. Recent approaches could outperform humans, and this problem seems nearly solved for common object scenarios. These research findings were utilized for more complex tasks like image segmentation or object detection [10].

Image classification focuses primarily on color images. Color images often consist of three channels (red, green, and blue) and cover only the visible spectrum of light. Therefore, they can mimic human perception.

Hyperspectral recordings approximate the spectrum for each pixel of the image. Therefore, the number of channels is increased (to around 200), and the range of recorded wavelengths is extended. The additional wavelengths can carry information, which is helpful for complex classification tasks. These systems can perform tasks that aren't possible with pure human perception, allowing superhuman performance in several applications.

The evolution of hyperspectral camera systems decreased their price and simplified their usage. This enabled their application in many new areas, e.g. food processing and agriculture [21][32], medical applications [16], inline sorting [28], besides their original use in remote sensing [1].

The application of hyperspectral cameras often requires complex data acquisition (e.g., line-scan operation mode) and labeling procedures. This is leading to small data sets. The small data sets and the complicated features, which are often necessary for the tasks, support overfitting. Besides these characteristics, the larger channel dimension of hyperspectral recordings requires special attention. To tackle this issue, using dimension reduction techniques, such as PCA [9] or Factor Analysis [29], as a preprocessing step are very

A PyTorch implementation of the model is available at:
https://github.com/cogsys-tuebingen/hyve_conv

common. These methods aim at removing redundant information and lead to more meaningful data.

A further problem arises from the fact that the recordings created by cameras of different manufacturers are a priori not compatible. There is no standardization regarding the distribution of the channels in the wavelength space. Therefore, two near-infrared cameras from different manufacturers, which cover the same spectral range, have generally different wavelength assignments for the channels. A model, which identifies the features based on the channel index, will fail on the recordings of another camera. In general, a solution for this problem is standardizing the recording to defined wavelengths. A basic and reliable approach is linear interpolation [27].

In this work, we want to tackle the two mentioned problems and propose a modified 2D convolution layer optimized for hyperspectral recordings. We outperform comparable approaches and reduce the parameters significantly by inferring a proximity bias for the channel dimension. Further, our model can incorporate the channels' wavelength information. This capability allows the training of camera-agnostic models, meaning the models can perform their tasks on recordings of different cameras. Besides the theoretical background, we prove our claims with empirical experiments on two hyperspectral applications.

Our main contributions are:

- We analyze two challenges of hyperspectral recordings. Based on the challenges, we infer a construed bias for hyperspectral models.
- A 2D convolution optimized for hyperspectral recording is proposed. The convolution supports camera-agnostic behavior, is interpretable regarding the selected spectral features, and reduces the number of parameters.
- The proposed method is validated in two different hyperspectral applications with publicly available data sets.

2. Related Work

Classical machine learning approaches, like SVMs [6] or k-nearest-neighbors [20], are still very common for the classification of hyperspectral recordings. In recent years, deep-learning-based methods could outperform these in many hyperspectral applications. Chen *et al*. were one of the earliest adopters of deep learning for hyperspectral recordings [4]. Their approach was based on a PCA followed by stacked autoencoders and a final logistic regression. More recent methods utilize convolution layers and are therefore called convolutional neural networks. These can incorporate the spatial information of the neighboring pixels beside the spectrum of a single pixel. This additional information and the higher complexity of the methods supported their breakthrough. In this work, we focus on convolutional neural networks.

Convolutional neural networks can be divided into methods based on 2D convolutions, 3D convolutions, or a mixture of both. 2D convolutions perform only spatial convolutions. So the exchange of information between channels is limited and primarily conducted by the final fully connected layers. Makantasis *et al*. were the first who utilized 2D convolutions for hyperspectral recordings [17]. 2D convolutions are still very common for hyperspectral recordings[30], because they have less trainable parameters and can still incorporate spatial information.

In contrast, 3D convolutions can perform convolutions in all three dimensions of the hyperspectral cube. So they can incorporate additional information, but are also parameter hungry. Large models are hard to train on the small hyperspectral data sets. Therefore, many approaches try to optimize the model power-size-ratio. Smaller models with the same performance are preferred because they tend to overfit less and often produce more stable results over different training runs. Hamida *et al*. used 3D convolutions to classify hyperspectral remote sensing data [2]. He *et al*. introduced a multiscale 3D convolutional neural network, which applies 3D convolutions with different kernel sizes. This boosts the performance of the 3D convolutions. Roy *et al*. proposed Fusenet, which fuses the output of 3D convolutions by using residual fuse blocks [25].

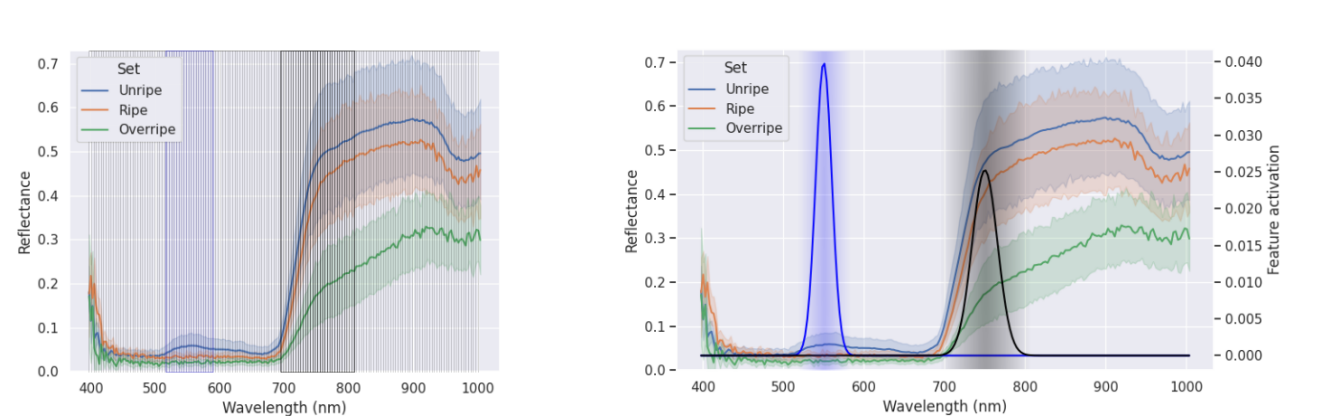
The third category combines 3D convolutions and 2D convolutions. Roy *et al*. proposed HybdriSN [25]. This model has a 3D convolution backbone. The output of this backbone is processed by a 2D convolution and a fully connected head.

SpectralNET [3] belongs to the first category, but it mimics 3D convolutions with wavelet transformations; therefore, it is also part of the third category.

Vision transformers, the most recent computer vision trend, also impacted the HSI classification. There are already some adaptations for hyperspectral recordings [23, 14]. But for this application, the transformer models can often only perform comparably performance to convolutional neural networks. Therefore, the smaller convolutional neural networks are often preferred.

Our method is based on 2D convolutions and, therefore, part of the first group. We utilize the wavelength meta information of the input channels to learn a continuous representation of the features in the input channel dimension. As our approach only affects the first convolution layer, it is compatible with other works mentioned.

Our method utilizes Gaussian distributions to represent the feature distribution in the input channel dimension. Still, our approach is not related to Bayesian convolutional neural



(a) Conv2D (b) HyveConv (ours)

Figure 2: The channel dimension handling visualized for normal 2D convolutions and our approach. Two example features are presented. Instead of learning each channel separately, the ranges of the features are learned.

networks [26]. These networks try to approximate the true posterior and incorporate the uncertainty into the inference process, which is not part of our approach.

Hu *et al*. proposed a similar approach [15]. Their Squeeze-and-Excitation block allows the network to learn a channel interdependency. Our approach differs in three key points. First, their method uses the input to predict weightings for each feature channel. Our method uses meta information, the channels' wavelength, to weigh the kernels. Further, our convolution introduced the bias that channels with similar wavelengths should use similar kernels. This proximity relation is helpful for hyperspectral records, shown in section 3.1. Last, our method also allows the interpretation of the learned features. The selected wavelengths can be visualized and analyzed, as shown in section 5.

Both methods share the idea of introducing an interdependency in the channel dimension. In the experiments, we can also show that our approach outperforms their approach in the hyperspectral application.

3. Proposed Method

The approach is based on 2D convolutions. Regular 2D convolutions handle input based on the input channel, which is not optimal for hyperspectral recordings. We emphasize key problems and justify our modifications. Further, we propose the method itself. The procedure is evaluated in the section afterward with experiments on two hyperspectral data sets with different applications.

3.1. Motivation

The reflected light, recorded by a camera, is a spectrum of many wavelengths. An RGB image oversimplifies this spectrum with three sampling bands (red, green, and blue). Hyperspectral recordings record many more bands and mimic a much better spectrum approximation. Fig. 2a shows the spectra of avocados with different ripening states recorded with HSI. The continuity of the underlying spectrum is captured sufficiently.

As a result, we encounter the two problems mentioned in the introduction. First, the numerous channels of the input data (around 200 bands) would demand a large first convolution layer. For the proposed method, the network should have access to the entire hyperspectral cube without a dimension reduction as preprocessing. A dimension reduction could reduce the size of the hyperspectral cube. But by using a dimension reduction, the original data can only be approximated. We argue that if the network can use the full potential of the data, this usually is beneficial for the selected features, as deep learning approaches can handle high dimensional data well [13]. We prove this in the experiment empirically (in section 4.1).

The second problem is that the recorded wavelengths of hyperspectral cameras are not standardized. Recordings of a manufacturer's near-infrared (NIR) camera are usually not compatible with recordings of a NIR camera of another manufacturer, even though both cameras share the same wavelength range. Their channel-wavelength assignments are often shifted and have different gradients. An example can be found in the supplementary material. As 2d-convolutions are based on the index of channels, standardizing the data by a preprocessing step, like linear interpolation [7], is necessary. These preprocessing steps harm the end-to-end training. We propose a method capable of handling different hyperspectral cameras by design.

To solve the mentioned problems, our convolution learns a wavelength range of interest (WROI) for each feature instead of the specific input channel. By having a continuous representation of the channel dimension, it can sample the

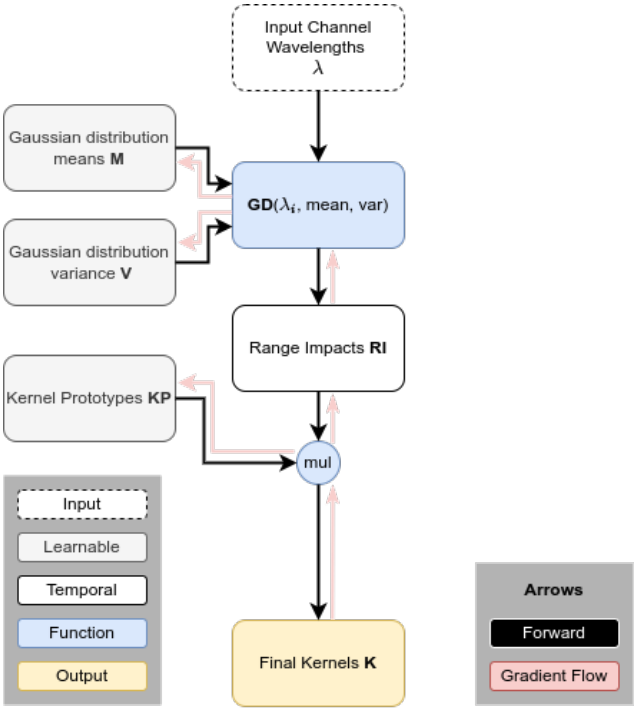


Figure 3: Flow of Hyperspectral Visual Embedding Convolution (HyveConv)

kernels based on the wavelengths of the input channels.

By adding the bias, that the network should apply similar kernels for similar wavelengths, it is possible to reduce the number of parameters significantly. This bias restricts the freedom of the model. Still, in the context of a continuous spectrum in the channel dimension, this is reasonable and seems one key point for handling hyperspectral recordings.

In summary, we propose a method that adds a bias regarding neighboring channels. It eliminates the need for dimensionality reduction for hyperspectral images. And it enables the training of hyperspectral camera-independent models. In section 4, we provide empirical evidence for these claims. But first, the method itself is described.

3.2. Method

The fundamental idea of this approach is to learn kernels and their target wavelength range in combination (Fig. 2b) instead of learning each input channel kernel independently (Fig. 2a). A learnable Gaussian distribution represents a wavelength range. The weighting factor for the corresponding kernel for this input channel is given by sampling the distribution at the input channel wavelength. The resulting kernel is then calculated by multiplying the factor and the kernel. Finally, the kernel is used for a 2D convolution on the specific input channel. Fig. 3 shows the method.

In the following, the method is described in further detail. Afterward, an extension is explained, which adds more synergy effects for the kernels.

A 2D convolution calculates the cross-correlation between trainable kernels and the input data. For C_{in} input channels, C_{out} output channels and kernel size of $K_x \cdot K_y$, this results in a matrix W for the trainable weights:

$$W \in \mathbb{R}^{C_{in} \times C_{out} \times K_x \times K_y} \tag{1}$$

The number of trainable parameters of a convolution depends on the number of input channels. For the first layer, the input channels are defined by the channel dimension of the input data. For hyperspectral recordings, this is around 200. For depthwise-separable convolutions [5] this relation is weakened but still exists (further analysis in the supplementary material).

To tackle the issue of a too large first layer, we learn wavelength ranges of interest (WROIs) for the kernels instead of channel-wise kernels. We assume that adjacent channels of the wavelength space typically share similar behavior. An example of this behavior can be found in Fig. 2a, where adjacent channels have very similar reflectance values, originating from the high resolution in the wavelength dimension and the continuity of the signal. Both points can be expected for HSI, so the bias to use similar kernels in neighboring bands seems suitable and even crucial.

Our convolution decouples the number of kernels from the number of input channels. Instead of kernels per input channel, wavelength ranges of interest (WROI) and their kernels are learned. G defines the number of WROIs. Their learnable kernels are called kernel prototypes (KP). This results in the matrix for the kernel prototype weights:

$$KP \in \mathbb{R}^{G \times C_{out} \times K_x \times K_y} \tag{2}$$

For the learnable distributions, Gaussian distributions (GDs) are used. A Gaussian distribution (GD) can mimic the wanted behavior, that the impact decays to the borders of a WROI. Further, it is defined by just two parameters, the mean μ and the variance σ^2. Both parameters are differentiable and interpretable. A Gaussian distribution is the combination of a learnable mean μ and a learnable variance σ^2. The value of the Gaussian distribution for a value x is defined as Eq. 3.

$$GD(x, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x-\mu}{2\sigma^2}\right) \tag{3}$$

To predict the kernels for specific channel wavelengths λ, the first step is to calculate the Gaussian distributions at these wavelengths λ. The result is the range impact matrix RI, which defines the impact of all kernel prototypes on all input channels:

$$\begin{aligned} &RI \in \mathbb{R}^{C_{in} \times G} \\ &\text{with } RI_{ij} = GD(\lambda_i, \mu_j, \sigma_j^2) \end{aligned} \tag{4}$$

Table 1: Parameters of a single convolution for the configuration: $C_{in} = 200$, $C_{out} = 25$, $K_x = 3$, $K_y = 3$, $G = 5$

Conv2D	**Depthwise -separable Conv2D**	**HyveConv (ours)**	**HyveConv++ (ours)**
$200 \cdot 25 \cdot 3 \cdot 3$ $= 45000$	$200 \cdot 1 \cdot 3 \cdot 3$ $+200 \cdot 25 \cdot 1 \cdot 1$ $= 6800$	$5 \cdot 25 \cdot 3 \cdot 3$ $+2 \cdot 5$ $=$ **1135**	$5 \cdot 25 \cdot 3 \cdot 3$ $+2 \cdot 5$ $+1 \cdot 25 \cdot 3 \cdot 3$ $+1 \cdot 1 \cdot 3 \cdot 3$ $+2$ $=$ **1371**

Afterward, the learnable kernel prototypes can be weighted with this matrix to produce the final kernels K. These kernels are then used for a 2D convolution on the input.

$$K = RI \cdot KP \in \mathbb{R}^{C_{in} \times C_{out} \times K_x \times K_y} \tag{5}$$

The result of this convolution is input to further layers. Our convolution can reduce the trainable parameters to W_{hyve} with $G << C_{in}$. In Tab. 1 the number of trainable parameters of the different models is compared.

$$W_{hyve} = \left(KP, M \in \mathbb{R}^G, V \in \mathbb{R}^G_{>0}\right) \tag{6}$$

To support full end-to-end learning, a gradient for Gaussian distributions and kernel prototypes is needed. Fig. 3 shows the gradient flow in our convolution model. The multiplication divides the gradient of the final kernel K on the learnable kernel prototypes KP and the range impact matrix RI. The matrix RI holds entries, which were sampled, of the Gaussian distributions regarding the wavelengths of the input. This allows us to infer the impact of the input wavelengths on the gradient. Further, the gradient can be passed to the means M and variances V of the Gaussian distributions. The channel wavelengths λ are part of the input data and do not need a gradient. So, all learnable components of the convolution are trainable based on the gradient of the final kernel K, and end-to-end training of the model is possible.

In the supplementary material, the behavior of the introduced hyperparameter G is discussed in further detail.

3.2.1 Extension: Additional Kernel Sharing

The learned WROIs allow the model to share kernels through the channel dimension of the input C_{in}.
As an extension, we propose sharing parts of kernels through the channel dimension of the output C_{out} and overall kernels of the convolution layer. For this, the previous method is enhanced with additional kernel prototypes. These additional kernel weights are weighted with the learnable factors $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}$. The sum of all kernel prototypes is then used further.

$$\begin{aligned} KP_{++}^{i,j,m,n} &= KP^{i,j,m,n} + \alpha \cdot KP_{c\_out}^{1,j,m,n} + \beta \cdot KP_{conv}^{1,1,m,n} \\ KP_{c\_out} &\in \mathbb{R}^{1 \times C_{out} \times K_x \times K_y} \\ KP_{conv} &\in \mathbb{R}^{1 \times 1 \times K_x \times K_y} \end{aligned} \tag{7}$$

The kernel prototypes KP_{++} replace the kernel prototypes KP in Eq. 5 resulting in Eq. 8, which predicts the final kernels.

$$K_{++} = RI \cdot KP_{++} \in \mathbb{R}^{C_{in} \times C_{out} \times K_x \times K_y} \tag{8}$$

With this extension, our approach has the following trainable parameters:

$$W_{hyve++} = (KP, M, V, \alpha, KP_{c\_out}, \beta, KP_{conv}) \tag{9}$$

With this extension, the model can utilize the synergy effects within the kernels of a convolution layer better.

It is important to keep the impact of the shared kernel prototypes at the beginning small. Otherwise, the training is very unstable. Therefore, we recommend an initial value for α_0 and β_0 of 0.1. An evaluation of the impact of the proposed extension can be found in the supplementary material.

4. Experiments

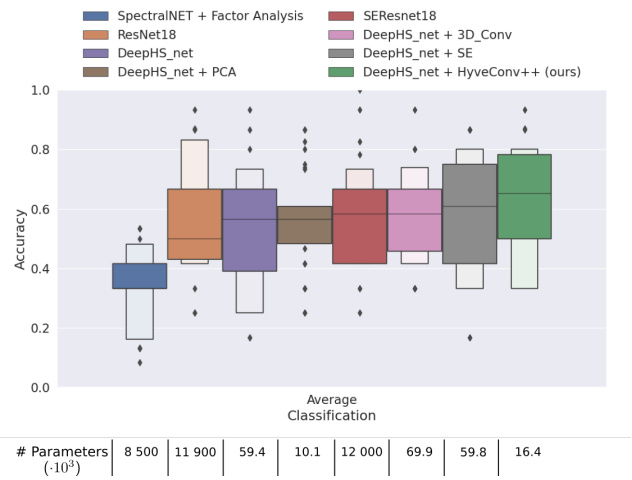
The proposed method is evaluated in two hyperspectral applications in the following section. The first application covers a classification task of ripening fruit recorded under laboratory conditions [30]. The second covers a well-established segmentation task of remote sensing data recorded by satellites.

For the following experiments, the extended version of the proposed method (see Eq. 8) with the following parameters was used: $G = 5$, $\alpha_0 = 0.1$ and $\beta_0 = 0.1$. Each composition was tested with three random seeds. The random seed affects the network initialization, the training sample order, and the data augmentation order.

4.1. Application A: Fruit Ripeness Prediction

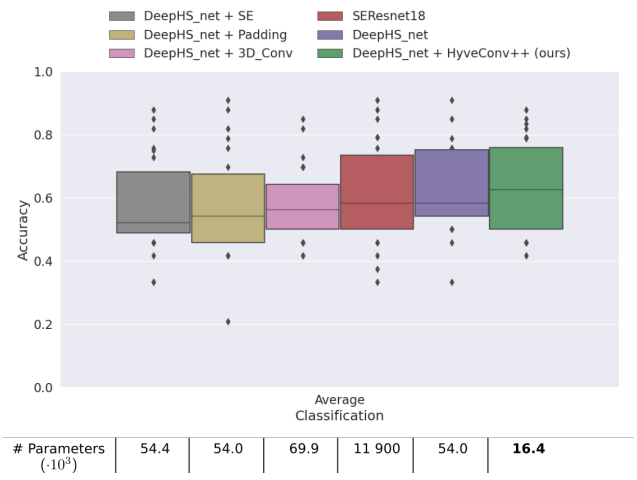
The first application's task is to classify fruit's ripeness level. The used data set [30] for this application is one of the largest labeled hyperspectral data sets publicly available. In addition, there are nearly no data set with recordings of different hyperspectral cameras for the same scene available. This property is relevant for testing the camera-agnostic behavior of our model.

Data Set The data set covers four fruit types (avocado, kiwi, persimmon, and mangos). Ripeness is classified into three categories (firmness, sweetness, and overall ripeness). Avocados have no sweetness; therefore, these only cover



# Parameters ($\cdot 10^3$)	8 500	11 900	59.4	10.1	12 000	69.9	59.8	16.4

Figure 4: Overall accuracy on the ripening fruit data set with the Specim FX 10 recordings.



# Parameters ($\cdot 10^3$)	54.4	54.0	69.9	11 900	54.0	**16.4**

Figure 5: Overall accuracy on the ripening fruit data set with recordings of both cameras.

the first and last categories. This sums up to eleven different setups (e.g., the firmness prediction for mangos or the sweetness prediction for kiwis). All setups cover three classes (unripe, perfect, and overripe). This data set has fixed training and test sets. Most of the recordings were done by a Specim FX 10. This hyperspectral camera covers the wavelength range of $397.66nm$ to $1003.81nm$ with 224 bands. In addition, there are many recordings of a Corning microHSI 410 Vis-NIR Hyperspectral Sensor. It covers the wavelengths between $408.03nm$ and $901.26nm$ with 249 bands. All recordings are already normalized with a white and a dark reference.

We use the training and test pipeline proposed in [30].

Models For the experiments, we used four models. As a basis for our approach, the model DeepHS_net[30] is used. It is a shallow convolutional neural network consisting of three depthwise-separable convolutions, a global average pooling layer, and a fully connected head. The complexity of the model is low, which helps in understanding the model internals. Further, the model could already achieve satisfying results for the prediction of the ripeness level of fruit. It is optimized for the small size of hyperspectral data sets.

For our model, we replaced the first convolution layer with a HyveConv++ layer, keeping all dimensions of the convolution fixed. The rest of the model stays the same. The training schedule of the original publication is used.

Further, we used a ResNet18 [11], which is still a commonly used backbone, and SpectralNET with factor analysis [3], which achieves state-of-the-art performance on the remote sensing data set. Finally, we used a Squeeze-and-Excitation (SE) [15] in two variants. First, in combination with a ResNet18 network and second with the DeepHS_net model. The SE block adds interdependency in the channel dimension of the kernels, which is similar to our approach and therefore interesting for comparison.

One Camera In the first experiments, the recordings of the Specim FX 10 were used only. These experiments indicate the performance of the tested models in the default single-camera use case. The results are visible in Fig. 4. Our model could outperform the other models' average accuracy, given the median over all categories and fruit types. A DeepHS_net with Squeeze-and-Excitation blocks (DeepHS_net + SE) performed second best. Both approaches introduce an interdependency between the channels, which seems helpful. Our model bias that similar wavelengths should have similar features further boosts performance. As a result, our model produced better results with fewer parameters.

Multiple Cameras In the second set of experiments, recordings of two different cameras were used (Specim FX 10 and Corning microHSI 410 Vis-NIR). These cameras differ in the recorded wavelengths. Further analysis can be found in the supplementary material. These experiments validate the camera-agnostic behavior. The number of recordings of each camera is not balanced ($\approx 2:1$), which adds another challenge. In these experiments, the training and test set contained recordings of both cameras.

We compared our method with the best models of the previous experiment. Only the HyveConv++ can process recordings with different wavelength-channel assignments by design. For the other models, preprocessing is necessary. The most basic approach is the usage of zero padding in the

Table 2: Classification accuracies (%) of different models in terms of overall accuracy (OA), Cohen Kappa (Kappa), and averaged classwise accuracy (AA) with 10% and 30% annotated data as training data, respectively. Based on the evaluations of Chakraborty *et al*. [3]. Two configuration of our model are presented. (*) not fine-tuned (**) larger hidden layers.

Training Samples	**Methods**	**# of params**	Indian Pines dataset			Pavia University dataset			Salinas dataset		
			OA	Kappa	AA	OA	Kappa	AA	OA	Kappa	AA
10%	SVM [6]		81.67±0.7	78.76±0.8	79.84±3.4	90.58±0.5	87.21±0.7	92.99±0.4	94.46±0.1	93.13±0.3	93.01±0.6
	2D-CNN [18]	561,300	80.27±1.2	78.26±2.1	68.32±4.1	96.63±0.2	95.53±0.2	94.84±1.4	96.34±0.3	95.93±0.9	94.36±0.5
	3D-CNN [2]	991,596	82.62±0.1	79.25±0.3	76.51±0.1	96.34±0.2	94.90±1.2	97.03±0.6	85.00±0.1	83.20±0.7	89.63±0.2
	M3D-CNN [12]	372,544	81.39±2.6	81.20±2.0	75.22±0.7	95.95±0.6	93.40±0.4	97.52±1.0	94.20±0.8	93.61±0.3	96.66±0.5
	FuSENet [24]	100,880	97.11±0.2	97.25±0.2	97.32±0.2	97.65±0.3	97.69±0.3	97.68±0.4	99.23±0.1	99.97±0.2	99.16±0.1
	HybridSN [25]	5,122,176	98.39±0.1	98.16±0.1	98.01±0.2	**99.72±0.1**	**99.64±0.1**	99.20±0.1	**99.98±0.2**	**99.98±0.2**	**99.98±0.1**
	SpectralNET [3]	6,800,336	**98.76±0.2**	98.59±0.1	98.61±0.1	99.71±0.1	99.62±0.1	99.43±0.2	99.96±0.2	99.96±0.1	99.97±0.1
	HyveConv++ (ours) *	**16,700**	98.18±0.6	98.41±0.1	98.28±0.1	99.30±0.3	99.30±0.3	**99.49±0.2**	99.24±0.2	99.64±0.0	99.94±0.0
	HyveConv++ (ours) **	**25,200**	98.33±0.1	**98.64±0.1**	**98.69±0.1**	99.42±0.2	99.49±0.2	99.46±0.2	99.89±0.0	99.79±0.0	99.74±0.0
30%	SVM [6]		87.24±0.4	85.27±0.5	85.15±1.1	95.65±0.1	94.63±0.2	94.60±0.1	94.95±0.1	94.48±0.1	97.93±0.1
	2D-CNN [18]	561,300	88.90±1.3	87.01±1.6	85.70±1.0	96.50±0.4	96.55±0.3	96.00±0.1	96.75±0.6	96.71±0.7	98.57±0.2
	3D-CNN [2]	991,596	90.23±0.2	89.70±0.3	89.87±0.1	97.90±0.3	97.22±0.1	97.30±0.1	95.54±0.5	94.81±0.3	97.09±0.6
	M3D-CNN [12]	372,544	95.67±0.1	94.70±0.3	94.60±0.6	97.60±0.2	96.50±0.6	98.00±0.1	94.99±0.3	95.40±0.1	96.28±0.2
	FuSENet [24]	100,880	99.01±0.2	98.60±0.1	98.64±0.1	99.42±0.2	99.21±0.3	99.33±0.2	99.68±0.2	99.74±0.1	99.69±0.1
	ImprovedTransformerNet [23]	150,000,000	99.22 %	99.19 %	99.08 %	99.64 %	99.49 %	99.67 %	99.91 %	99.78 %	99.63 %
	HybridSN [25]	5,122,176	99.75±0.1	99.71±0.1	99.63±0.2	99.98±0.1	**99.98±0.2**	99.97±0.2	**100**	**100**	**100**
	SpectralNET [3]	6,800,336	**99.86±0.2**	**99.84±0.2**	**99.98±0.1**	**99.99±0.1**	**99.98±0.1**	**99.98±0.1**	**100**	**100**	**100**
	HyveConv++ (ours) *	**16,700**	99.85±0.0	99.75±0.0	99.7±0.2	99.97±0.0	99.96±0.0	99.97±0.0	99.98±0.0	99.99±0.0	99.99±0.0
	HyveConv++ (ours) **	**25,200**	**99.86±0.0**	**99.84±0.0**	99.57±0.1	99.96±0.0	**99.98±0.0**	99.94±0.0	99.93±0.0	99.92±0.0	99.99±0.0

channel dimension to produce recordings of the same size. A more advanced method is linear interpolation, which is often the first choice for this problem. For linear interpolation, the configuration is crucial. An inappropriate definition of the quantization steps can harm the expressiveness of the recordings. Linear interpolation was the default preprocessing and was used for most of the models.

The results are visible in Fig. 5. Our method could outperform the other approaches in the average accuracy and the model size. Linear interpolation could improve the performance in these experiments. The Squeeze-and-Excitation approach, which performed well in the previous experiment, did not improve the performance of the models here.

By using the wavelength information for the convolution, HyveConv++ can boost the camera-agnostic behavior for HSI.

An additional experiment, which supports this claim, is provided in the supplementary material.

4.2. Application B: Satellite Imagery Segmentation

Data Set The second application, Hyperspectral Remote Sensing Scenes (HRSS), is a collection of hyperspectral satellite images collected by M. Graña, M.A. Veganzons, and B. Ayerdi. The task is to classify the nature of the ground in different settings. Each scene consists of an image with ground truth labels. The most common scenes are Indian Pines (IP), Pavia University (UP), and Salinas (SA). Each scene is handled separately. We followed the data handling procedure of Chakraborty et al. [3]. The segmentation task is converted into a classification task of 64x64 patches.

Indian Pines was recorded with the AVIRIS sensor, which covers the range of $400nm$ to $2500nm$ with 224 channels [1]. Twenty-four noisy channels were already removed from these recordings. The classification happens within 16 classes. Salinas was also recorded by this sensor and covers six classes. Pavia University was recorded with the ROSIS sensor, covering the range from $430nm$ to $830nm$ with 103 bands and distinguishing nine classes.

Models We used two configurations of our model. Both used the whole hyperspectral cube without dimension reduction. The first configuration (*) is the same model used for the ripening classification task. Only the final output layer was adapted to the number of classes.

The second configuration (**) was slightly adapted for this application. The main change here was to increase the number of channels in the hidden layers, reasoned by the higher number of classes for this application.

Results Tab. 2 provides an overview of the performance of the models in the remote sensing application. We tested a variety of different models. The models cover classical machine learning [6], 2D convolutions [18], 3D convolutions [2], mixture of 2D and 3D convolutions [12, 24], 2D convolutions with optimized preprocessing [25, 3] and one vision transformer based approach [23]. Especially, HybridSN [25] and SpectralNET [3] achieve state-of-the-art results for this application. The model without modifications (**) could already achieve second/third rank in the overall list (see Tab. 2). With minor modifications (**), it even achieved state-of-art performance with 250 times fewer parameters. These experiments show that our proposed method generalizes well to different hyperspectral applications.

5. Ablation Study

In the previous section, our model outperformed comparable models in two applications. In this section, the interpretability of the learned WROIs, also called camera filters, is presented. For an in-depth ablation study, we refer to

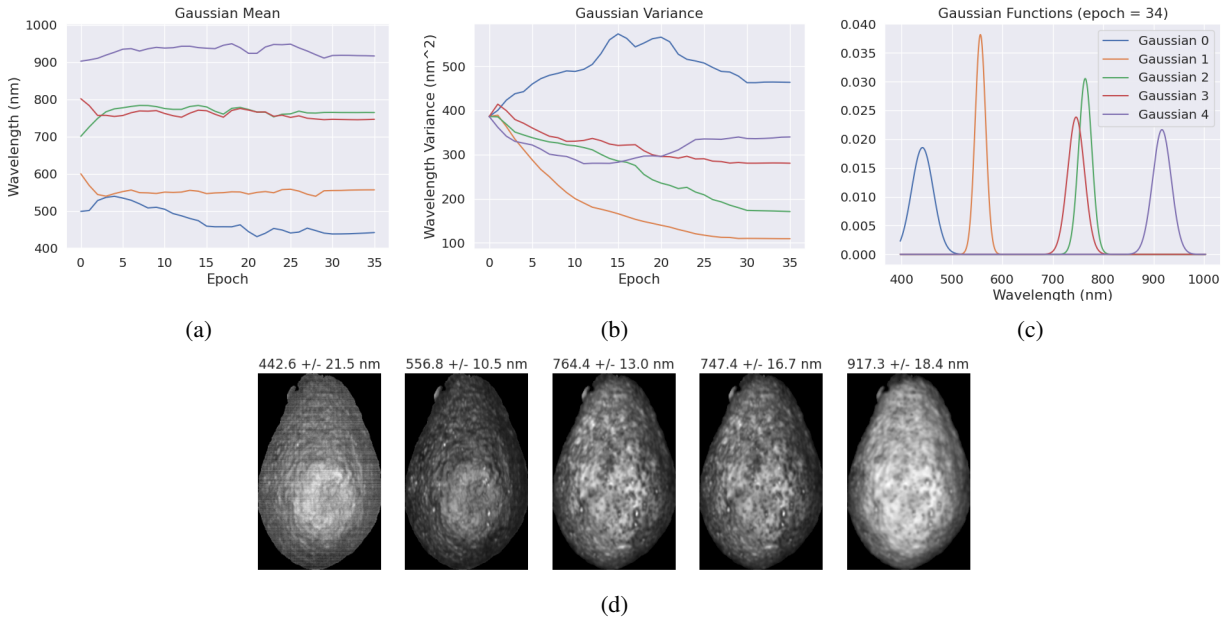


Figure 6: Training of the Gaussian distributions. (a) and (b) show the development of the mean and variance over training epochs. (c) shows the final Gaussian distributions, and these are applied as filters in (d).

the supplementary material. It contains an analysis of the influence of the hyperparameter G, defining the number of WROIs. Further, the proposed extension of the method (Eq. 9) is analyzed. And it is checked whether the training of the Gaussian distribution is necessary.

Fig. 6 presents the training and the resulting WROIs for an example run for the ripeness classification of avocados. 6a and 6b show the value over epochs for the mean and the variance, respectively. The variance is a good indicator for the WROI search procedure. A decreasing variance indicates that the model found a feature and narrows the corresponding wavelength range. The final WROIs are visible in Fig. 6c and the visualizations of these are shown in Fig. 6d. Gaussian distributions 2 and 3 have a final overlap of over 50%. They even had a position swap in epoch 3. Therefore, they may cover the same feature, and a reduction of the number of the WROI G could be possible. The final camera filters cover the visible light in the ranges of blue and green. Further, overtones of water and the region, which indicate the degeneration of chlorophyll [31], were selected. These ranges fit the findings of previous works [22, 19]. The WROI selection of the trained model could be used to build a multispectral camera optimized for this use case. A multispectral camera with 5-10 custom-defined wavelengths is normally easier to apply in an inline scenario.

We showed that the learned features are explainable and can even be used further.

6. Conclusion

In this work, we proposed a 2D convolution optimized for hyperspectral recordings. By using a continuous representation of the kernels and adding a suitable model bias, it is possible to significantly reduce the number of parameters. Further, sampling the kernels by the wavelengths of the input allows the training of a camera-agnostic model. The whole model is end-to-end trainable. And it introduces only one interpretable hyperparameter G. The parameter defines the number of wavelength ranges of interest, also called camera filters. Experiments on two different hyperspectral applications could confirm the advantage of this method.

The convolution is proposed for hyperspectral imaging. Still, it could also be helpful in other scenarios of image data with many channels and some proximity relationship between them. This is part of future research.

Multispectral and color cameras are typically based on wavelength filters. The WROIs, learned by our method, are these kinds of filters. Therefore, the model learns the necessary filters for a specific task in a data-driven way. These could be used to build an optimized multispectral camera.

Acknowledgment

The German Ministry for Economic Affairs and Energy has supported this work, Project Avalon, FKZ: 03SX481B. The computing cluster of the Training Center Machine Learning, Tübingen has been used for the evaluation, FKZ: 01IS17054.

References

[1] Marion F. Baumgardner, Larry L. Biehl, and David A. Landgrebe. 220 band aviris hyperspectral image data set: June 12, 1992 indian pine test site 3, Sep 2015.

[2] Amina Ben Hamida, Alexandre Benoit, Patrick Lambert, and Chokri Ben Amar. 3-d deep learning approach for remote sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 56(8):4420–4434, 2018.

[3] Tanmay Chakraborty and Utkarsh Trehan. Spectralnet: Exploring spatial-spectral waveletcnn for hyperspectral image classification. *arXiv preprint arXiv:2104.00341*, 2021.

[4] Yushi Chen, Zhouhan Lin, Xing Zhao, Gang Wang, and Yanfeng Gu. Deep learning-based classification of hyperspectral data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(6):2094–2107, 2014.

[5] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1800–1807. IEEE Computer Society, 2017.

[6] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[7] P.J. Davis. *Interpolation and Approximation*. Dover Books on Mathematics. Dover Publications, 1975.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[9] Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 770–778. IEEE Computer Society, 12 2016.

[12] Mingyi He, Bo Li, and Huahui Chen. Multi-scale 3d deep convolutional neural network for hyperspectral image classification. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3904–3908, 2017.

[13] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[14] Danfeng Hong, Zhu Han, Jing Yao, Lianru Gao, Bing Zhang, Antonio Plaza, and Jocelyn Chanussot. Spectralformer: Rethinking hyperspectral image classification with transformers. *IEEE Trans. Geosci. Remote. Sens.*, 60:1–15, 2022.

[15] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(8):2011–2023, 2020.

[16] Guolan Lu and Baowei Fei. Medical hyperspectral imaging: a review. *Journal of Biomedical Optics*, 19(1):1 – 24, 2014.

[17] Konstantinos Makantasis, Konstantinos Karantzalos, Anastasios Doulamis, and Nikolaos Doulamis. Deep supervised learning for hyperspectral data classification through convolutional neural networks. In *International Geoscience and Remote Sensing Symposium (IGARSS)*, volume 2015-Novem, pages 4959–4962. Institute of Electrical and Electronics Engineers Inc., 11 2015.

[18] Konstantinos Makantasis, Konstantinos Karantzalos, Anastasios Doulamis, and Nikolaos Doulamis. Deep supervised learning for hyperspectral data classification through convolutional neural networks. In *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 4959–4962, 2015.

[19] Angela Melado-Herreros, Sonia Nieto-Ortega, Idoia Olabarrieta, Mónica Gutiérrez, Alberto Villar, Jaime Zufía, Nathalie Gorretta, and Jean-Michel Roger. Postharvest ripeness assessment of 'hass' avocado based on development of a new ripening index and vis-nir spectroscopy. *Postharvest Biology and Technology*, 181:111683, 2021.

[20] Antonio Mucherino, Petraq J. Papajorgji, and Panos M. Pardalos. *k-Nearest Neighbor Classification*, pages 83–106. Springer New York, New York, NY, 2009.

[21] B. Park and R. Lu. *Hyperspectral Imaging Technology in Food and Agriculture*. Food Engineering Series. Springer New York, 2015.

[22] Jhon Pinto, Hoover Rueda-Chacón, and Henry Arguello. Classification of hass avocado (persea americana mill) in terms of its ripening via hyperspectral images. *TecnoLógicas*, 22(45):111–130, 2019.

[23] Yuhao Qing, Wenyi Liu, Liuyan Feng, and Wanjia Gao. Improved transformer net for hyperspectral image classification. *Remote. Sens.*, 13(11):2216, 2021.

[24] Swalpa Kumar Roy, Shiv Ram Dubey, Subhrasankar Chatterjee, and Bidyut Baran Chaudhuri. Fusenet: fused squeeze-and-excitation network for spectral-spatial hyperspectral image classification. *IET Image Process.*, 14(8):1653–1661, 2020.

[25] Swalpa Kumar Roy, Gopal Krishna, Shiv Ram Dubey, and Bidyut B. Chaudhuri. Hybridsn: Exploring 3-d-2-d CNN feature hierarchy for hyperspectral image classification. *IEEE Geosci. Remote. Sens. Lett.*, 17(2):277–281, 2020.

[26] Kumar Shridhar, Felix Laumann, and Marcus Liwicki. A comprehensive guide to bayesian convolutional neural network with variational inference. *CoRR*, abs/1901.02731, 2019.

[27] J.F. Steffensen. *Interpolation: Second Edition*. Dover Books on Mathematics. Dover Publications, 2013.

[28] Petra Tatzer, Markus Wolf, and Thomas Panner. Industrial application for inline material sorting using hyperspectral imaging in the nir range. *Real-Time Imaging*, 11(2):99–107, 2005. Spectral Imaging II.

[29] L.L. Thurstone. *The Vectors of Mind: Multiple-factor Analysis for the Isolation of Primary Traits*. University of Chicago science series. University of Chicago Press, 1935.

[30] Leon Amadeus Varga, Jan Makowski, and Andreas Zell. Measuring the ripeness of fruit with hyperspectral imaging and deep learning. In *International Joint Conference on Neural Networks, IJCNN 2021, Shenzhen, China, July 18-22, 2021*, pages 1–8. IEEE, 2021.

[31] Alan R. Wellburn. The spectral determination of chlorophylls a and b, as well as total carotenoids, using various solvents with spectrophotometers of different resolution. *Journal of Plant Physiology*, 144(3):307–313, 1994.

[32] Hai Qing Yang. Nondestructive prediction of optimal harvest time of cherry tomatoes using vis-nir spectroscopy and plsr calibration. In *Emerging Engineering Approaches and Applications*, volume 1 of *Advanced Engineering Forum*, pages 92–96. Trans Tech Publications Ltd, 9 2011.