IMAGE COMPRESSION USING VECTOR-QUANTIZED AUTO ENCODERS WITH SEMANTICALLY MEANINGFUL FEATURE EXTRACTION

Bart Beusen (1), Xenia Ivashkovyc (2), Tanja Van Achteren (3)

(1) *Flemish institute for technological Research (VITO), tel. +3214335843, bart.beusen@vito.be*
(2) *Flemish institute for technological Research (VITO), tel. +3214336874, tanja.vanachteren@vito.be*
(3) *Flemish institute for technological Research (VITO), tel. +3214335929, xenia.ivashkovych@vito.be*

ABSTRACT

Modern earth observation missions collect imagery with ever more spatial and spectral detail, which causes the amount of data to increase very rapidly. Limitations in data downlink capacity have become a major bottleneck hindering the exploitation of the rich information in current and future satellite missions. In the CORSA project, an ESA PhiLab EO Science for Society project, we developped an AI method for lossy image compression which learns optimal data reduction tuned to the desired quality for the intended use. It generates a compact image representation containing deep feature vectors that are part of a codebook. The vectors can be represented by their codebook indices, which can be written to file as compact bit-arrays, suitable for downlinking. The full vectors are recreated afterwards from the indices using the same codebook, and the image can be reconstructed from the vectors using the decoder part of the model. The method is very suitable for satellite onboard data reduction, especially for smaller remote sensing missions capable of acquiring large volumes of multispectral and/or high spatial resolution image data, but limited in power and downlink budget. The compressed format can also be used to store image data efficiently in the ground segment.

1. INTRODUCTION

Modern earth observation missions capture ever more detailed information, with more spatial and spectral detail. Consequently, the amount of data gathered is increasing very rapidly. The use of smaller platforms like cubesats also limits data downlink capacity, hindering usability of the missions. Also for the ground segments, the increase in data amount poses new challenges regarding data storage capacity and fast image retrieval.

Diverse solutions are being developed to tackle this problem. The idea of "edge computing" is to make computations close to the source to avoid unnecessary large data transfers. Applied to satellite imaging, performing onboard processing and data analysis offers a fundamental shift as the derived information to be downlinked is much more compact. The success of onboard processing solutions is limited by computational resources, including a strict power budget which is shared with imaging and downlink activities. Solely downlinking extracted information instead of the raw data also puts a limit on the potential uses cases.

Onboard data reduction can be achieved by detecting and discarding clouded images (which are useless for most applications). This functionality has very recently been showcased in the ϕ-Sat [1], or PhiSat, AI technology which will fly on one of the two CubeSats that make up the FSSCat mission. To avoid downlinking cloudy images back to Earth, the ϕ-Sat artificial intelligence chip will filter them out so that only usable data are returned. The Φ-sat-2 mission will embark a SW platform able to run AI apps to fully decouple the App features from the underlying on-board hardware. opening up a more flexible path for on-board AI applications.

Generic lossless compression techniques only offer modest gains, therefore “near-lossless” methods, which retain the full quality of raw images but not the exact bits, may be more beneficial for certain applications. Even greater data reduction gains can be obtained by taking into account the purpose of the images. Classical lossy image compression (e.g. JPG) are optimized for human viewing and minimize the visibility of compression artefacts. However, resulting images are less suitable to derive quantitative information afterwards. It can be much more advantageous to adapt the compression to minimize the loss in terms of the quality of the information that will be derived from the images. In this paper, we investigate the use of artificial neural networks for image compression with semantically meaningful compressed representations.

Recently, neural networks, especially the convolution neural networks (CNN), have achieved significant success in many fields including computer vision [2]. A CNN model can be interpreted as feature extractors to transform the image and video into feature space with compact representation, which is beneficial for image and video compression. One important approach in using Neural Networks for image compression, is the use of auto-encoder architectures, consisting of an encoder mapping an image input through nonlinear transforms to a latent feature vector (typically with lower dimensionality than the original input image), and a decoder which inverts the latent feature back into an image reconstruction. Once the encoder outputs the latent feature, typically some form of binarization/quantization is performed in order to transform the latent feature into a set of discrete symbols. The quantization contributes to information loss, but by discretizing the feature space, we get a finite set of symbols that we can further compress into a bitstream using some lossless entropy coding function. The encoder and decoder can be constructed using convolutional neural networks, or using recurrent neural networks [3, 4].

However, it is difficult to straightforwardly incorporate the CNN model into end-to-end image compression. Generally speaking, CNN training depends on the back-propagation and stochastic gradient descent algorithm which demand the almost-everywhere differentiability of the loss function with respect to the trainable parameters such as the convolution weights and biases. Due to the quantization module in image compression, it produces zero gradients almost everywhere which stops the parameters updating in the CNN.

A variational autoencoder (VAE) is used in the work “Variational image compression with a scale hyperprior” [5]. The paper proposes an end-to-end trainable model for image compression based on variational autoencoders. Unlike existing autoencoder compression methods, their model trains a complex prior jointly with the underlying autoencoder. They demonstrate that this model leads to state-of-the-art image compression when measuring visual quality using the popular MS-SSIM index, and yields rate–distortion performance surpassing published ANN-based methods when evaluated using a more traditional metric based on squared error (PSNR).

Vector-Quantized Variational Autoencoders (VQVAE) provide an unsupervised model for learning discrete representations by combining vector quantization and autoencoders [6]. As [5]**,** VQ-VAE also trains a complex prior jointly with the underlying autoencoder. VQVAE-2 introduced a multi-scale hierarchical organization of VQ-VAE for the purpose of generative AI, improving also the compression performance [7].

Next to the self-supervised objective of image reconstruction, we also try to incorporate a self-supervised objective that is targeted specifically at learning semantically meaningful laten representations. We choose to use DINO or “self-**di**stillation with **no** labels” [8]. With DINO no negative samples are needed as in self-supervised methods that use Contrastive predictive loss. Just like BYOL [9], DINO uses 2 networks, i.e. student and teacher networks, where the weights of the teacher network are not learned by backpropagation, but are an exponential moving average of the weights from the student network (so they lag behind the student's weight parameters). DINO relies heavily on cropping and different data augmentation techniques to feed different versions of the same input image to the teacher and student networks

We developed an autoencoder model for image compression using the quantization method as implemented in the VQVAE-2 architecture, and added the DINO approach in the training procedure. Our method is very suitable for satellite onboard data reduction, especially for smaller remote sensing mission which target specific application or application areas. The compressed format can also be used to store image data in the ground segment. Not only does this optimizes the use of the available data storage, it also allows to perform many image analysis tasks directly on the compressed image vectors, which can be more efficient than on the original images.

The compression quality is assessed using Structural Similarity Index (SSIM) and Peak Signal-to-Noise Ratio (PSNR) on compressed Sentinel-2 images. The effectiveness of using the latent features as direct input to the downstream task of classification was evaluated using the BigEarthNet (http://bigearth.net) dataset [10]. The effect of the compression on the downstream task of semantic segmentation was evaluated on the task of detecting agricultural parcel boundaries using as input images reconstructed from the compressed representations using the decoder.

2. COMPRESSION METHOD AND RESULTS

2.1 Compression Network architecture

The proposed method of this paper follows the concept of the hierarchical VQVAE-2 [7], with some changes in the model architecture to achieve higher reconstruction accuracies. The model is an autoencoder, consisting of an encoder and a decoder pathway. The encoder generates a compact image representation containing deep feature vectors that are part of a learned codebook. The vectors can be represented by their codebook indices, which can be written to file as compact bit-arrays, suitable for downlinking. The full vectors are recreated afterwards from the indices using the same codebook, and the image can be reconstructed from the vectors using the decoder part of the autoencoder model.

Our encoder consists of one or more convolutional blocks, each one downsampling its input's width and height with a factor of two. Each convolutional block in our encoder produces a feature map consisting of a set of feature vectors that are further quantized based on their distance to the prototype vectors in the codebook, such that each vector is replaced by the index of the nearest prototype vector in the codebook. For each convolutional block in its encoder, our model produces a quantized feature map or 'latent representation" with a high and width depending on the level in out convolutional hierarchy. For a three-level encoder, the model outputs three latent representations, i.e. "top latent", "middle latent" and "bottom latent".

In the decoder part of the model, every quantized feature map is mapped back to their corresponding full vector using the learned codebook. The "top latent" and "middle latent" are first up-sampled to match the width and height of the "bottom latent". From these vectors the decoder then reconstructs the original input image using transposed convolutions. To learn these mappings, the gradient of the reconstruction error is back-propagated through the decoder, and to the encoder using the straight-through gradient estimator as described in [6].

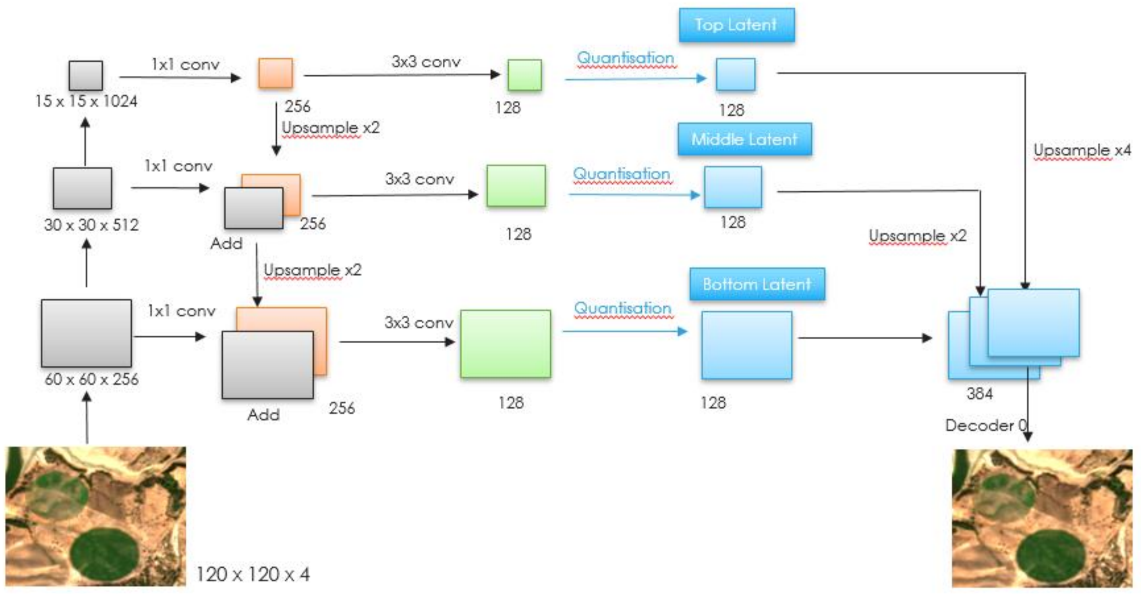

Figure 1. Overview of The VQVAE-FPN model architecture

In this work, several variations to the VQVAE-2 architecture were tested to achieve the best trade-off between compression ratio and reconstruction accuracy for Sentinel-2 image patches from BigEarthNet, only taking into account the 10m bands (B02, B03, B04, B08) for every image. Image size is 120 by 120 pixels.

The design choices for our experiments were

- number of hierarchic levels in our VQVAE model: 1, 2 or 3
- embedding dimension: dimension of the latent vectors in the codebook
- number of embeddings: the number of possible vector values in the codebook to which the latent vectors will be mapped; a larger number of embeddings will increase the capacity of the information bottleneck, but will lead to a lower compression ratio as we will need more bits to store the index-number of each codebook vector.
- Number of convolutional filters used in each encoder block and in the residual blocks.

The encoder in each block is the “top” encoder from the [7], performing only 1 downsampling operation. In contrast to the original VQVAE-2 paper, we do not perform a 4x downsampling of the input image in the first encoder block, but we limit this to 2x downsampling.

Bottom level feature maps are conditioned on non-quantized higher feature maps, using upsampling for the higher feature maps instead of using a dedicated decoder. This conditioning is performed by a simple addition instead of a concatenation. A 3x3 convolution is applied to the conditioned feature maps on all levels to reduce the number of feature maps to the match the embedding dimension. The decoder takes as input the quantized latent feature maps from all hierarchical levels, upsampled to the resolution of the lowest latent feature map. We refer to this architecture as “VQVAE-FPN”, since the alterations are loosely based on some specific architectural design aspects from Feature Pyramid Networks. The resulting model is described in figure 1.

2.2 Dataset

The model for image compression was trained on images from the BigEarthNet dataset [10]. BigEarthNet is a benchmark archive, consisting of 590,326 pairs of Sentinel-1 and Sentinel-2 image patches. For our study, we only used the Sentinel-2 images. In the BigEarthNet dataset, each image was annotated by the multiple land-cover classes (i.e., multi-labels) that were provided from the CORINE Land Cover database of the year 2018 (CLC 2018).

Representation of the 43 available labels over all images in the dataset is very unbalanced. Some labels only occur in as few as 300 images, while other labels are present in up to 22k images. For the compression testcases, a subset of 75k images was made, selected with a bias in favor of the rarer classes so that our dataset was more balanced over the classes.

The Sentinel-2 images are encoded using a uint16 format, with most values being lower than 8000. Thus, pixel values are mostly situated in the lower part of the dynamical range. To be used as input for the neural network, pixel values were first normalized using scikit-learn power scaler, leading to a more normalized distribution of input values.

The model is applied to Sentinel-2 10m bands (Red, Blue, Green, NIR), and performances are validated in terms of reconstruction quality and effect on downstream tasks of image classification and semantic segmentation.

2.3 Reconstruction results

Table 1 shows the reconstruction accuracies for this VQVAE-FPN with feature map depths of 128, 256 and 512 for bottom (level 0), middle (level 1) and top level (level 2) feature maps respectively. The size of the codebook for each quantization level, i.e. the number of embeddings, is indicated in the table as “NE0”, “NE1” and “NE2” for each hierarchical level respectively. The number of bytes needed to store the codebook index numbers to file, are shown in columns “bytes bin0”, “bytes bin1” and “bytes bin2” for each level respectively. The number of bytes required may be calculated as indicated in Eq. 1

$$\text{number of bytes} = 1/8*(w*h*\log2(NE)) \tag{1}$$

with w, h and NE the width, height and “Number of Embeddings” respectively for the relevant feature map. In the case we would like to save the quantized feature map of level 0 using 8-bits per index, our codebook for the bottom level feature map will have a size of 2^8 = 256 possible vectors. The resulting size for the compressed features of the bottom feature map would be 1/8*(60*60*8) = 3600 bytes. The original uncompressed images are encoded using a uint16 format. A raw RGB-NIR image (4 channels) of size 120 by 120 pixels thus has a size of 115200 bytes (120*120*4*(16/8)). As a baseline, jpeg compression on RGB (no NIR) images with a compression ratio of 22 led to an average SSIM score of 0.951.

Model name	NE0	NE1	NE2	PSNR	SSIM	MSE	bytes bin0	bytes bin1	bytes bin2	Compression ratio
S2-128-256 Upsample	256	256		69,11	0,943	0,011	3600	900		25.60
S2-128-256 Upsample	512	512		69,82	0,951	0,010	4050	1013		22.75
S2-128-256 Upsample	1024	1024		70,89	0,960	0,008	4500	1125		20.48
S2-128-256-512 Upsample	256	256	256	69,80	0,950	0,010	3600	900	225	24.38
S2-128-256-512 Upsample	512	512	512	70,05	0,953	0,009	4050	1013	254	21.67
S2-128-256-512 Upsample	1024	1024	1024	70,56	0,959	0,008	4500	1125	282	19.50

Table 1. Model description and reconstruction metrics for 6 different model versions

For all test on downstream tasks, we used model "S2-128-256-512 Upsample" with NE=1024 because a 3-level hierarchy produces a top latent feature map that may be used directly as input to a classifier model. Only for compression of the images consisting of the six 20m Sentinal-2 bands, a 2-level hierarchy was used, "S2-128-256 Upsample", also with NE=1024, to compensate for the smaller image input size of 60x60 pixels compared to 120x120 pixels for the 10m bands B02, B03, B04 and B08.

3. DOWNSTREAM APPLICATIONS

We investigate the effect of image compression on two downstream tasks: semantic segmentation using compressed-reconstructed images, and binary classification using the intermediate feature maps as direct input to a small multilayer perceptron that acts as the classifier.

3.1 Semantic segmentation on reconstructed images

Methodology

We set up an experiment to test the impact of image quality loss caused by the compression-reconstruction on the downstream task of image segmentation. For this testcase, data from the AI4EO challenge was used [11]. The goal of this challenge is to map cultivated land using Copernicus Sentinel imagery, and to develop solutions to extract as much information as possible from the native 10-meter per pixel resolution. To test the effect of artefacts caused by the image compression on the quality of downstream tasks, this AI4EO challenge is very well suited since the aim of the challenge was to identify agricultural areas smaller or narrower than a Sentinel-2 pixel. More specifically, the goal was to estimate a cultivated land binary map at 2.5 meter spatial resolution given as input a Sentinel-2 time-series at 10 meter spatial resolution, therefore resulting in a 4x spatial resolution enhancement. The participants were requested to produce output datasets on one area of interest in Slovenia. The size of the input images was 500 by 500 pixels, or 5 by 5 km².

To perform the segmentation, we used the trained model that was submitted to the challenge by the "AI@TAP" team. Their model is a Super-Resolution Conditional GAN inspired from the Pix2Pix Image-to-Image Translation with Conditional Adversarial Networks paper [12]. The model needs as input a timeseries of Sentinel-2 images, using 10 of the 12 bands available: B02, B03, B04, B05, B06, B07, B08, B8A, B11, and B12. In the original dataset, all bands were resampled to a ground resolution of 10m.

The trained model for semantic segmentation was tested on the original input data, as well as on input data that was compressed and reconstructed using our VQVAE-FPN approach. The compression of the original AI4EO Challenge dataset was performed using 2 separate VQVAE-FPN models: one model specifically trained on the BigEarthNet 10m Sentinel-2 bands (B02, B03, B04 and B08), and another model trained on the BigEarthNet 20m Sentinel-2 bands (B06, B07, B08A, B11, and B12). The compression model for the images with only the 20m bands uses an encoder with only 2 blocks instead of 3, since the input image size in the BigEarthNet training set for these bands is 60x60 pixels instead of 120x120 pixels as for the 10m bands.

Since in the AI4EO dataset the native 20m bands were resampled to 10m, the images in our testset for the 20m bands were first resampled to their original 10m ground resolution before running the compression with the VQVAE-FPN model. After compression, the reconstructed output was again upsampled from 20m to 10m ground resolution so it could be used as a direct replacement for the data in the original AI4EO dataset.

Results

We compare the results of the segmentation by running the same segmentation model on three versions of the same input:

1. Original AI4EO Challenge input data
2. Compressed-reconstructed data using VQVAE-FPN-256 model for the 10m bands
3. Compressed-reconstructed data using VQVAE-FPN-1024 model for the 10m bands

Reconstruction accuracies for all three testcases are shown in table 2. From this table we see a minimal impact on accuracy, precision and recall when using the compressed-reconstructed data compared to using the original input data. We notice no significant difference between using data compressed with the VQVAE-FPN-256 (256 embedding vectors in the codebook) compared to data compressed with the VQVAE-FPN-1024 (1024 embedding vectors in the codebook).

Input data	**Accuracy**	**Precision**	**Recall**
AI4EO original	0.942	0.898	0.898
VQVAE-FPN-256 compressed-reconstructed	0.936	0.890	0.882
VQVAE-FPN-1024 compressed-reconstructed	0.936	0.890	0.884

Table 2. Results on semantic segmentation using three different sets of input data: original AI4EO data, compressed-reconstructed data using VQVAE-FPN with 256 and 1024 embedding vectors respectively.

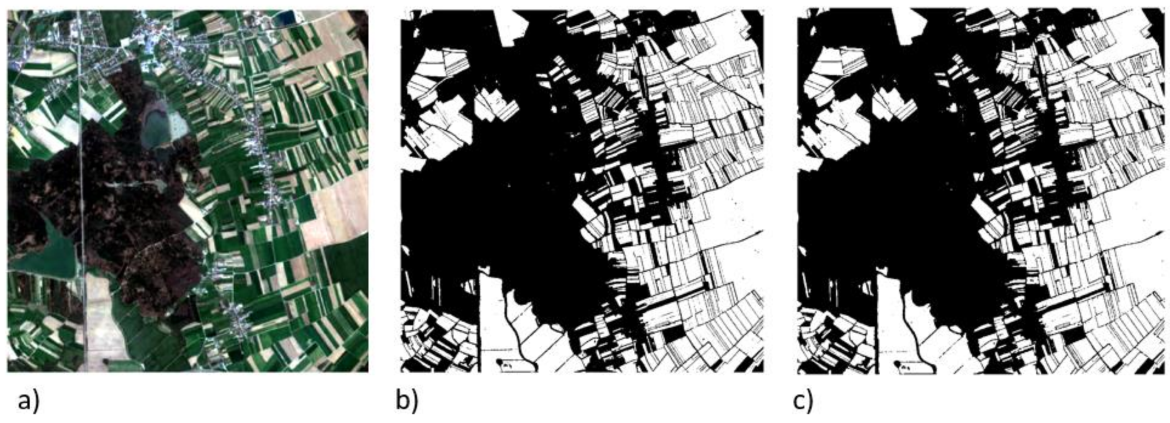

Figure 2. a) RGB subset of the original 10-bands input image. b) output of the semantic segmentation model applied to the original input image; c) output of the semantic segmentation model applied to the VQVAE-FPN compressed-reconstructed input image

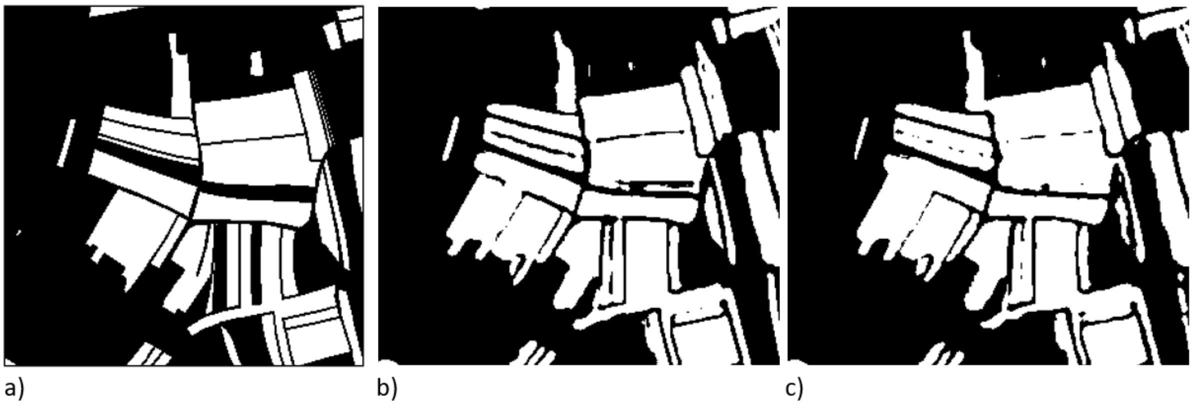

Figure 3. Detail of the segmentation results. a) ground truth map of cultivated land b) output of the semantic segmentation model applied to the original input image; c) output of the semantic segmentation model applied to the VQVAE-FPN compressed-reconstructed input image

From the table 2 and figure 2, one may conclude that the compression-reconstruction of the input images using VQVAE-FPN does have a negative impact on the accuracy of the detected cultivated land in the super-resolved output maps although this impact is very small (0.006). The main impact is seen in the very fine details (output map is super-resolved to 2.5m ground resolution i.e. 1/16th of a Sentinel-2 pixel that has 10m ground resolution) such as fine borders in between two detected regions. Figure 3 shows a detail of the output map that clearly shows the differences in some of the finer structures.

3.2 Classification using VQVAE-FPN intermediate features

A convolutional classification model typically consists of a convolutional backbone that produces a set of features that are forwarded to a "classification top", i.e. a one or more fully connected layers on top of the convolutional backbone that output the prediction for each class.

A possible benefit of image compression with the VQVAE-FPN model, is that the features extracted by the encoder backbone may be used as direct input for a classification top. In contrast to a classical supervised classification pipeline, where encoder and classification top are trained end-to-end with backpropagation using a supervised objective, our workflow would allow to train only the classification top fully connected layers using the (quantized) VQVAE-FPN features as direct input.

Methodology

We investigate if the VQVAE-FPN encoder backbone can be used as an efficient feature extractor for classification of Sentinel-2 images from the BigEarthNet dataset. We add a fully connected classifier on top of the VQVAE-FPN encoders, taking as input either the non-quantized top latent feature map from the encoder backbone, or the quantized latent feature maps. The multilayer perceptron used as a classifier consists of a global average pooling operation, followed by 2 dense layers with 512 hidden neurons, ReLU activation and dropout applied after each layer, a 'layer normalization' layer, and finally an dense layer with one neuron with a sigmoid activation function. For training the Adam optimizer was used in combination with binary cross-entropy loss.

The autoencoder model should be generic enough to allow not only high quality image reconstruction, but also to capture the features necessary for downstream applications. For this purpose, we add the self-supervised training of our encoder network using DINO as a pre-task, before the model is further finetuned for image compression using the image reconstructions as the self-supervised objective. We implemented DINO in Tensorflow, using 2 global crops and 6 smaller crops of each image as input to the student network, whereas the only the 2 global crops are input to the teacher network.

It is expected that DINO self-supervised pretraining yields a encoder backbone that is able to produce semantically meaningful features, whereas this would not be the case when training with only the compression-reconstruction objective.

For the binary classification testcases, special care was given to construct well balanced datasets using a subset of the BigEarthNet dataset. The size of each subset was defined by the minimum of images in each class that was part of the testcase. For every positive example in the binary classification dataset for a specific class, the exact same number of negative examples was selected. Datasets for binary classification were created for

- Coniferous forest vs. broad-leaved forest
- Sea vs no-sea
- Urban fabric (continuous and discontinuous) vs non-urban areas
- Presence of waterbodies vs no waterbodies.

To check the effect of the DINO pretraining on the classification accuracy, and to check the effect of quantization of the feature maps on the classification accuracy, the following test methodology was applied:

1. Optional: DINO pre-training
    a. Create the encoder-only model
    b. self-supervised training for the encoder model using DINO
2. Construct full VQVAE-FPN model, optionally using pre-trained encoders from step 1
3. Repeat 5 times:
    a. Test VQVAE-FPN on reconstruction accuracies
    b. Freeze the VQVAE-FPN backbone
    c. Test binary classification on the 4 datasets by training the fully connected classifier on top of the feature maps of the frozen encoder backbone for 50 epochs with 100 steps per epoch
    d. Test binary classification on the 4 datasets by training the fully connected classifier on top of the **quantized** feature maps from the frozen VQVAE-FPN model for 50 epochs with 100 steps per epoch
    e. Unfreeze the VQVAE-FPN model and train for 5 epochs on the reconstruction task

The methodology is thus applied two times: once using a VQVAE-FPN with randomly initialized encoders (no DINO pre-training in step 1), and once using a VQVAE-FPN with the DINO pre-training of step 1.

Effect of reconstruction training on the quality of the features for classification

In this testcase, the encoder is not pre-trained using DINO. The VQVAE-FPN is trained for the reconstruction task, and every 5 epochs, the encoder part of the VQVAE-FPN is frozen so its intermediate feature maps can be tested as input to a classifier on the binary classification task. As explained in the test methodology, for the classification task the encoder is thus frozen and only the classification top is trained. Two versions of the encoder are used: one that outputs the raw feature maps (not quantized), and one version that outputs the quantized feature maps. In the classification task, classifiers were trained for 50 epochs and the accuracy reached at the end of the training is shown in the figures of Fig 4.

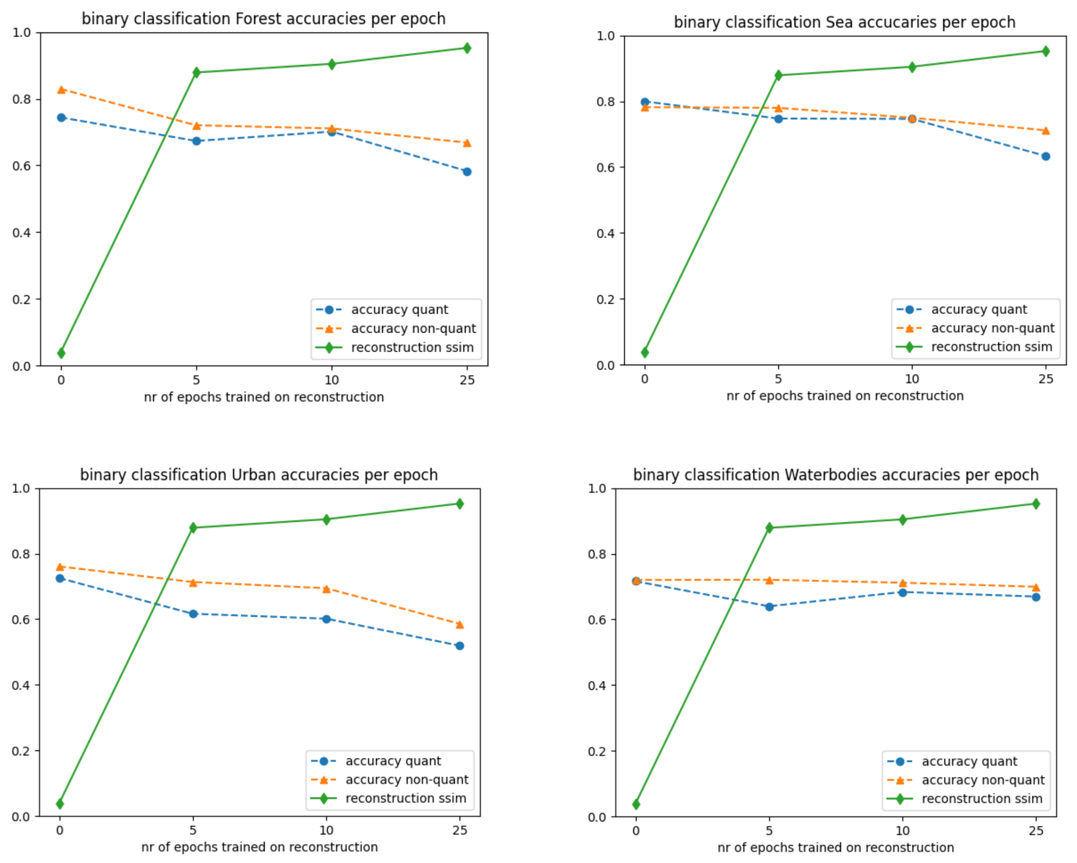

Figure 4. Results on 4 binary classification tasks and reconstruction accuracy, using the intermediate features from the encoder backbone as input for the classifier, tested at different times during the VQVAE-FPN training process on reconstruction (nr of epochs on the x-axis); results are shown both for quantized vs non-quantized intermediate feature maps as input to he classifier.

For each binary classifier, we see in figure 4 a decrease in classification accuracy as the encoder is finetuned further for the reconstruction task (increasing SSIM scores). We also notice that classification on the non-quantized top feature map consistently produces higher accuracy than classification on the quantized top-level feature map. When running the same testcases, but with DINO pre-training of the encoders (figure not shown), the same decrease in classification accuracy is noticed.

Effect of DINO pre-training on the semantic quality of the quantized feature maps

Figure 5 shows the results of the same testcases, presented in such a way as to visualize the effect of DINO pre-training. It shows the results for a classifier using as input the quantized feature maps from the frozen encoder. The results of this experiment show no beneficial effect of DINO pre-training. Also in the case of classification using non-quantized feature maps, no beneficial effect of DINO pre-training can be seen (figure not shown).

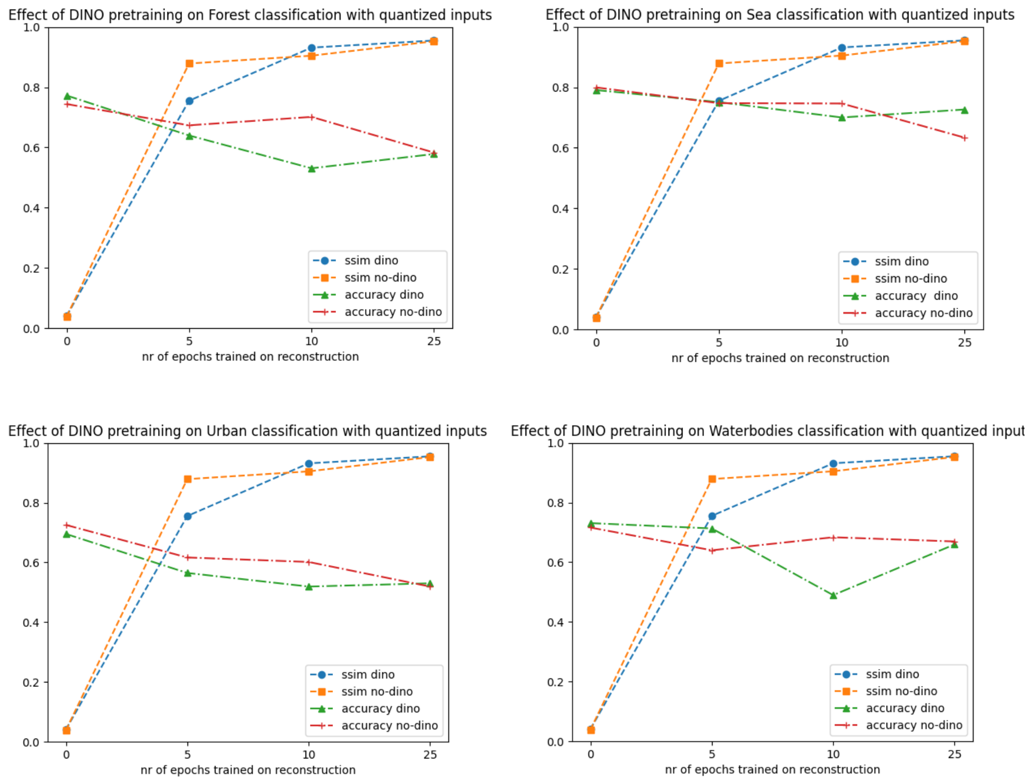

Figure 5. Results on 4 binary classification tasks and reconstruction accuracy, using the quantized intermediate features from the encoder backbone as input for the classifier, tested at different times during the VQVAE-FPN training process on reconstruction (nr of epochs on the x-axis), with DINO vs without DINO pre-training.

From these results, we may conclude that the DINO pre-training task did not have the beneficial effect that we strived for. In future work, we will investigate if DINO pre-training with smaller image size as input might lead to better results. With smaller input images, the variability withing one image is further reduced, which may lead to a better training signal for the DINO training task. Also the image augmentations used in our custom Tensorflow implementation of DINO might be too conservative compared to the augmentations used in the original DINO implementation.

End-to-end supervised classification baseline

We notice that training a classifier on a frozen encoder backbone that was randomly initialized (standard Tensorflow “Glorot Uniform” initialization), already provides useable features for the 4 different binary classification tasks. This can be seen by looking at the accuracies at “epoch 0” in the previous figures. At epoch 0 the encoder backbone was still randomly initialized as no training was performed at this stage. The training of the encoder on the reconstruction task led feature extractions that perform worse on the classification tasks.

As a baseline, we also ran a purely end-to-end supervised classification training using the encoder (no DINO pre-training) and the classifier model on top. In this testcase, the encoder was not pre-trained and the encoder weights were not frozen. We noticed that the training did only converge for the case of the “Sea” vs “No Sea” binary classification task, reaching an accuracy, precision and recall of 0.781, 0.812 and 0.743 respectively. For the other binary classification tasks the model failed to converge, indicating that the images in these datasets may contain too much variation (too many land use classes present in one image, waterbodies too small in comparison to image size) or unclear separation between two classes (e.g. no differences visible between broad-leaved and coniferous in a deforestated area). To our surprise, we must conclude that training a classifier with a purely supervised objective, using a classification head on top of a randomly initialized convolutional backbone with frozen weights, outperforms the end-to-end training where the weights of the encoder backbone are also updated during the training process. In our experiments, the end-to-end training led to an instable training process that was unable to converge in the case of three out of four binary classification tasks (Coniferous forest vs. broad-leaved forest, Urban fabric (both continuous and discontinuous) vs non-urban areas, Presence of waterbodies vs no waterbodies).

4. CONCLUSION

We presented an autoencoder model for image compression using the quantization method as implemented in the VQ-VAE-2 architecture, and added the DINO approach in the training procedure. We showed a small trade-off in image reconstruction accuracy vs compression ratio, while the scores on the Structural Similarity Index (SSIM) and Peak Signal-to-Noise Ratio (PSNR) were comparable to a jpeg2000 baseline with a comparable compression factor.

The impact of image quality loss caused by the compression-reconstruction was tested on the downstream task of image segmentation, using data from the AI4EO challenge [11] where the goal was to map cultivated land using Copernicus Sentinel imagery. We concluded that the compression-reconstruction of the input images using VQVAE-FPN did have a negative impact on the accuracy of the detected cultivated land in the super-resolved output maps although this impact was very small (0.006). The main impact was visible in the very fine details of the super-resolved output map.

To test the semantic quality of the encoded features of the quantized feature vectors, we set up an experiment where a classifier was trained using the encoded features (quantized or non-quantized) as direct input. A classifier was created for 4 different binary classification tasks based on different subsets of the BigEarthNet dataset [10]. For each binary classifier, we noticed a decrease in classification accuracy as the encoder is finetuned further for the reconstruction task (increasing SSIM scores). We also noticed that classification using the non-quantized top feature map consistently produced higher accuracy than classification using the quantized top-level feature map as input to the classifier. With the applicability for space missions in mind, this poses a trade-off between training for reconstruction accuracy vs. useability of the compressed features for classification. Furthermore, increasing the size of the codebook will lead to a smaller bottleneck in quantization and might close the gap between classification accuracy using quantized vs non-quantized feature maps. However this will come at the cost of a decreased compression factor.

To our surprise, our experiments showed that training a classifier using a classification head on top of a randomly initialized convolutional backbone with frozen weights, outperforms the end-to-end training where the weights of the encoder backbone are also updated during the training process.

Future research will focus on enhancing the pre-training task either using DINO or another self-supervised objective. Instead of training consecutively first on DINO and then on reconstruction, a combined training objective might lead to quantized feature maps with higher semantic quality. Since there is no theoretical restriction on the number of input channels in the input image, a possible follow-on opportunity would be to investigate the use of this method for hyperspectral missions.

5. REFERENCES

[1] https://philab.phi.esa.int/explore/the-phi-lab-explore-office/phi-sats-programme

[2] Krizhevsky, A., Sutskever, I., and Hinton, G. E. *ImageNet classification with deep convolutional neural networks*, NIPS, pp. 1106–1114, 2012.

[3] George Toderici, Sean M. O'Malley, Sung Jin Hwang, Damien Vincent, David Minnen, Shumeet Baluja, Michele Covell, Rahul Sukthankar. *Variable Rate Image Compression with Recurrent Neural Networks*, arXiv:1511.06085, 2016.

[4] Lucas Theis, Wenzhe Shi, Andrew Cunningham, Ferenc Huszár. *Lossy Image Compression with Compressive Autoencoders*, arXiv:1703.00395, 2017.

[5] Johannes Ballé David Minnen Saurabh Singh Sung Jin Hwang Nick Johnston. *Variational image compression with a scale hyperprior,* arXiv:1802.01436, 2018.

[6] A. Oord, K. Kavukcuoglu, and O. Vinyals. *Neural discrete representation learning*, Advances on Neural Information Processing Systems (NIPS), Long Beach, CA, Dec. 2017.

[7] A Razavi, A van den Oord, O Vinyals. *Generating diverse high-fidelity images with vq-vae-2*, Advances in Neural Information Processing Systems, 14837-14847, 2019.

[8] Caron et al., 2021. *Emerging properties in self-supervised vision transformers*, arXiv:2104.14294v1 [cs.CV] 29 Apr 2021.

[9] Jean-Bastien Grill, Florian Strub, Florent Altch´e, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, R´emi Munos, and Michal Valko. *Bootstrap your own latent: A new approach to self-supervised learning.* In NeurIPS, 2020. 2, 3, 4, 5, 8, 9, 10, 14, 15, 16, 18.

[10] G. Sumbul, M. Charfuelan, B. Demir, V. Markl. *BigEarthNet: A Large-Scale Benchmark Archive for Remote Sensing Image Understanding*, IEEE International Conference on Geoscience and Remote Sensing Symposium, pp. 5901-5904, Yokohama, Japan, 2019.

[11] https://platform.ai4eo.eu/enhanced-sentinel2-agriculture

[12] Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. *Image-to-Image Translation with Conditional Adversarial Networks*, In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 5967–76. Honolulu, HI: IEEE, 2017. https://arxiv.org/abs/1611.07004.