Deep learning for satellite image compression

Pascal Bacchus

COLLEGE MATHS, TELECOMS
DOCTORAL INFORMATIQUE, SIGNAL
BRETAGNE SYSTEMES, ELECTRONIQUE

Université de Rennes

THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES

ÉCOLE DOCTORALE N° 601
Mathématiques, Télécommunications, Informatique, Signal, Systèmes, Électronique
Spécialité : *Signal, Image, Vision*

Par

Denis Pascal BACCHUS

Deep learning for satellite image compression

Thèse présentée et soutenue à Rennes, le 12/12/2023
Unité de recherche : INRIA Rennes

Rapporteurs avant soutenance :

Marc ANTONINI — Research Director, CNRS I3S Nice-Sophia Antipolis
Marie CHABERT — Professor, INP Toulouse

Composition du Jury :

Président :	Olivier DEFORGES	Professor, INSA Rennes
Rapporteurs :	Marc ANTONINI	Research Director, CNRS I3S Nice-Sophia Antipolis
	Marie CHABERT	Professor, INP Toulouse
Examinateurs :	Olivier DEFORGES	Professor, INSA Rennes
	Joan SERRA-SAGRISTÀ	Professor, Universitat Autònoma de Barcelona
Dir. de thèse :	Aline ROUMY	Research Director, INRIA Rennes
Co-enc. de thèse :	Christine GUILLEMOT	Research Director, INRIA Rennes

Invité :

Renaud FRAISSE — Imaging Systems Senior Expert, Airbus Defence and Space, Toulouse

Table of Contents

Résumé en Français

Contexte

La nouvelle génération de caméras embarquées pour les satellites permet d'acquérir des images de plus grandes résolutions spatiales et spectrales. Cet accroissement de la résolution se traduit par une plus grande quantité de données à traiter à bord du satellite et à transmettre au sol. D'autant plus que ces dernières années ont vu une explosion de l'utilisation des données d'observation terrestre dans une multitude d'applications de la défense à l'agriculture en passant par l'étude du climat et de la biodiversité. La transmission de ce grand volume de données nécessite donc des solutions de compression efficaces et capables de préserver toutes les informations présentes dans l'image et nécessaires aux tâches d'analyse.

La solution de compression actuellement utilisée a été définie par le comité consultatif pour les systèmes de données spatiales (CCSDS) et repose sur une transformée en ondelettes [CCSDS, 2017]. Cette solution proche du standard JPEG 2000 [Skodras et al., 2001] concernant ses performances de compression a été développée en gardant à l'esprit un compromis entre les performances de compression et la complexité algorithmique de la méthode. En effet, cette problématique sur la quantité de données à transmettre se situe dans le contexte d'un système embarqué qui possède donc des contraintes physiques fortes sur les ressources de calcul disponibles.

Il est aussi nécessaire de replacer ce processus de compression dans l'ensemble plus large de la chaîne de traitement des systèmes d'imagerie qui va du capteur jusqu'à l'affichage.

En effet de nombreux traitements tels que le débruitage et le dématriçage sont nécessaires au sol pour restaurer les images. Dans les applications satellitaires, la compression est un maillon clé situé à l'interface entre l'espace et la Terre.

Objectifs de la thèse

L'objectif principal de la thèse est d'étudier de nouvelles techniques de compression offrant un meilleur compromis débit-distorsion en vue d'une utilisation à bord de nouveaux satellites à capteurs matriciels de haute résolution. Dans ce contexte, l'intérêt pour les méthodes de compression avec des réseaux de neurones profonds émerge en premier lieu. Celles-ci ont pu se développer grâce à trois facteurs travaillant de concert : l'explosion de la quantité de données, l'utilisation de matériel informatique hautement parallélisable (GPU) et le développement d'outils logiciels permettant de les utiliser. Ces méthodes ont démontré leur efficacité dans des champs très larges d'application y compris dans le domaine de l'imagerie et sont particulièrement performantes pour extraire les caractéristiques de données très typées.

Ces réseaux de neurones doivent cependant être adaptés pour les besoins de notre problème. Les images satellitaires ont la particularité d'avoir une entropie élevée due d'une part à leur résolution et d'autre part à la présence importante de détails haute fréquence e la taille d'un pixel. Le principal défi de la compression de ces images est de pouvoir distinguer, dans les informations à haute fréquence, la part du bruit de la part du signal pour une meilleure interprétation des données. Cela doit se faire en minimisant d'éventuels artefacts de compression pour ne pas détériorer les détails de l'image.

L'autre point crucial est la réduction de la complexité des réseaux de neurones afin de pouvoir les appliquer à des systèmes embarqués. Malgré l'augmentation des ressources au sein des satellites, les questions de mémoire et de complexité algorithmique restent bien présentes. Un enjeu de la thèse sera de concilier notre solution de compression avec d'autres traitements d'images pour profiter des capacités d'apprentissage de bout en bout des réseaux de neurones. Ces traitements combinés permettront de diminuer la charge de calcul en enlevant des étapes.

Structure de la thèse

Le manuscrit est organisé en deux parties précédées d'un premier chapitre qui rappelle les caractéristiques des systèmes d'imagerie satellitaire et ses traitements à bord. La première partie se concentre sur les questions de compression d'images RVB avec tout d'abord un état de l'art des méthodes d'apprentissage profond. Ces méthodes sont ensuite améliorées pour répondre aux enjeux d'une compression à bord. La deuxième partie est consacrée à l'extension de notre méthode de compression pour y inclure d'autres traitements tels que le débruitage et le dématriçage avec là aussi d'abord une mise en contexte des algorithmes pour effectuer des traitements joints suivis de nos contributions dans le contexte de l'imagerie satellitaire.

Dans le **Chapitre 1** nous détaillons l'ensemble de la chaîne de traitement pour une image satellite, de l'acquisition par les capteurs, à son débruitage et dématriçage au sol en passant par sa compression, étape nécessaire pour la transmission depuis l'espace jusqu'à la Terre. Cette base de traitements séquentiels au sein du satellite permet une mise en contexte pour la suite du manuscrit.

Le **Chapitre 2** présente la littérature des méthodes d'apprentissage profond pour les problèmes de compression d'images. Nous détaillons en particulier l'ensemble des réseaux de neurones utilisant des autoencodeurs et qui forment aujourd'hui l'état de l'art dans ce domaine. Ces réseaux sont conçus pour reproduire en sortie l'image d'entrée en extrayant les caractéristiques principales de l'image dans une représentation latente plus compacte. Cela en fait des réseaux particulièrement adaptés à notre problème et l'un d'eux, l'autoencodeur avec *hyperprior* [Ballé et al., 2018] sera le point de départ pour toute la suite du travail présenté.

Le **Chapitre 3** est notre premier chapitre de contributions. Nous y développons un réseau de neurones pour répondre à la compression d'images satellitaires RVB liées au déploiement de futur satellite avec des capteurs matriciels. Néanmoins ce type d'imagerie bien que RVB, repose sur des caractéristiques particulières bien différentes de l'imagerie naturelle et nécessite donc une adaptation de la méthode de compression à bord pour diminuer la bande passante afin de transmettre ces images au sol. Nous montrons l'efficacité de ces nouvelles méthodes utilisant l'apprentissage profond dans ce contexte d'images très

typées où ces réseaux réussissent à extraire les caractéristiques des images. Afin de rendre compatible l'algorithme compression aux conditions de système embarqué nous optimisons les paramètres du réseau (couches, filtres, noyaux) pour réduire le coût de calcul et chercher un meilleur fonctionnement à haut débit pour des qualités de reconstruction suffisantes. Nous ajoutons aussi la possibilité pour le réseau de compresser des images sur une large bande de débit avec un pas de quantification variable, au lieu d'être bloqué sur un point de débit-distorsion.

Les images satellitaires sur lesquelles nous travaillons possèdent une entropie importante, donc une quantité importante de détails dans les hautes fréquences. Pour améliorer à la fois la qualité de reconstruction générale et la capacité de compression des motifs striés de hautes fréquences, notre stratégie est d'identifier les zones haute fréquence. Pour ce faire, nous proposons l'ajout de modules d'attention au sein du réseau. Nous effectuons aussi un travail sur l'entraînement du réseau avec l'ajout d'une fonction de coût perceptuel au sein du compromis débit-distorsion. Notre fonction de coût perceptuel nous permet de mieux extraire lors de l'entraînement des zones jusqu'alors floutées par le schéma de compression optimisé sur une norme $L2$. Ce nouveau compromis fait lui aussi l'objet d'un travail avec l'optimisation des hyperparamètres du réseau grâce à une stratégie multi-objectifs automatique des paramètres.

Enfin, nous proposons une méthode de compression quasi sans perte avec l'ajout d'une compression spécialisée sur les hautes fréquences de l'image résiduelle. A travers l'utilisation d'un filtrage et seuillage de nos résidus, nous parvenons à extraire une information structurelle contenue dans l'erreur de reconstruction de notre réseau de compression générale.

À partir du **Chapitre 4**, tenant compte du déploiement récent et en cours de capteurs matriciels dans les satellites, et qui nécessitent une étape de dématriçage, nous proposons d'étendre notre méthode de compression pour ajouter d'autres traitements au sein du satellite. Dans ce chapitre, nous présentons l'état de l'art des méthodes de traitement conjoints pour les images en se concentrant uniquement sur le dématriçage, le débruitage et la compression qui sont les trois traitements au cœur de la chaîne de traitement classique en imagerie. Ces méthodes jointes permettent de gagner, tant sur la qualité de reconstruction que sur la réduction de complexité par rapport aux traitements séquentiels.

Dans le **Chapitre 5**, nous détaillons la fusion du débruitage et dématriçage au sein de notre réseau de compression. Ce traitement joint permet de répondre tant à des questions sur les performances de compression et reconstruction qu'à une problématique de complexité en diminuant le nombre de processus dans la chaîne de traitement. D'abord avec une preuve de concept du modèle qui mêle des données brutes et donc matricées avec un réseau de compression forçant une représentation couleur en sortie.

Cette base est améliorée avec l'ajout d'une branche de guidage pendant l'entraînement qui a pour but de forcer une plus grande représentation intermédiaire des caractéristiques de l'image. Cela s'adapte parfaitement à notre précédent schéma d'entraînement multi-objectifs sans provoquer une augmentation du coût de calcul à bord du satellite. Un pré-traitement des données brutes en amont du réseau est ajouté pour aider une meilleure reconstruction.

Pour être au plus proche de données réelles, nous considérons des données bruitées et procédons à une modification de l'entraînement du réseau pour effectuer un débruitage des images conjointement aux deux autres opérations. Enfin, une ouverture vers une méthode de compression à qualité constante (au lieu du débit constant) est envisagée dans le cadre de ces traitements joints. L'étude des statistiques de l'image en amont de la chaîne de traitement nous permet de faire une estimation des paramètres de qualité nécessaires pour assurer une compression ne dépassant jamais un certain seuil et donc assurer une qualité de reconstruction constante tout en minimisant le débit.

INTRODUCTION

Context

The new generation of cameras on board satellites can acquire images with greater spatial and spectral resolution. This increase in resolution means that more data must be processed on board the satellite and transmitted to the ground. That is especially true as recent years have seen an explosion in the use of Earth observation data for a wide range of applications, from defence and agriculture to the study of climate change and biodiversity. The transmission of these large volumes of data therefore requires efficient compression solutions capable of preserving all the information present in the image and necessary for the analysis tasks.

The compression solution currently used was defined by the Consultative Committee for Spatial Data Systems (CCSDS) and is based on a wavelet transform [CCSDS, 2017]. This solution, which is close to the JPEG 2000 standard [Skodras et al., 2001] in terms of compression performance, was developed with a compromise between compression performance and algorithmic complexity in mind. Indeed, the volume of data to be processed is an issue in the context of an embedded system, which has strong physical constraints on the available computing resources.

It is also necessary to consider this compression process in the processing pipeline of the imaging system, from sensor to display. On the ground, numerous processing operations such as denoising and demosaicking are required to restore images. In satellite applications, compression is an essential link at the interface between space and earth.

Objective of the thesis

The main objective of the thesis is to investigate new compression techniques that offer a better rate-distortion trade-off for use on board new satellites with high-resolution matrix sensors. In this context, there is a growing interest in compression methods using deep neural networks. These methods have been able to develop thanks to three factors: the explosion of data, the use of highly parallelisable processing hardware (GPUs), and the development of software tools to use them. These methods have demonstrated their effectiveness in a wide range of applications, including image processing, and are particularly effective at extracting features from highly specific data.

However, these neural networks have to be adapted to the needs of our problem. Satellite images have the particularity of having a high entropy due, on the one hand, to their resolution and, on the other hand, to the large amount of high-frequency pixel-sized detail. The main challenge in compressing these images is to be able to distinguish noise from signal in the high-frequency information, for a better interpretation of the data. This must be done while minimising any compression artefacts so as no to alter image detail.

The other key issue is to reduce the complexity of neural networks so that they can be applied to embedded systems. Despite the increase in resources on satellites, the issues of memory and algorithmic complexity are still very much present. One of the challenges of the thesis will be to combine our compression solution with other image processing techniques to take advantage of the end-to-end learning capabilities of neural networks. These joint methods will make it possible to reduce the computational load by removing steps in the imaging pipeline.

Outline of the thesis

The manuscript is divided into two parts, preceded by a first chapter that provides an overview of the satellite imagery system and its different processing tasks on board. The first part focuses on RGB image compression issues, starting with a state-of-the-art review of deep learning methods. These methods are then improved to meet the challenges of onboard compression. The second part is dedicated to the extension of our compression method to other processing operations such as denoising and demosaicking, again starting

with a description of the methods used for joint processing, followed by our contributions in the context of satellite imagery.

In **Chapter 1** we describe the entire image processing pipeline on board satellites, from acquisition by the sensors, to denoising and demosaicking on the ground. Compression as a necessary step for transmission from space to Earth is explained in detail. This overview of sequential processing within the satellite provides a context for the rest of the manuscript.

Chapter 2 presents the literature on deep learning methods for image compression problems. In particular, we detail neural networks using autoencoders that represent the state-of-the-art in this field today. These networks are designed to reproduce the output of the input image by extracting the main features of the image in a more compact latent representation. This makes them particularly well suited to our problem, and one of them, the *hyperprior* autoencoder [Ballé et al., 2018] will be the starting point for the rest of the work presented.

Chapter 3 is our first chapter of contributions. We develop a neural network to respond to the compression of RGB satellite images linked to the deployment of future satellites with matrix sensors. However, although this type of imagery is RGB, it is based on specific characteristics that are very different from natural images and therefore requires the onboard compression method to be adapted in order to reduce the bandwidth required to transmit these images to the ground. We demonstrate the effectiveness of these new methods using deep learning in the context of highly specific images, where these networks succeed in extracting the characteristics of the images. To make our compression algorithm compatible to the conditions of an embedded system, we are optimising the network parameters (layers, filters, kernels) to reduce the computational cost and achieve better performance at high bit rates with sufficient reconstruction quality. We are also adding the ability for the network to compress images over a wide bit rate range with a variable quantisation step, rather than being locked to a rate-distortion point.

The satellite images we are working on have a high entropy and therefore a large amount of detail at high frequencies. To improve both the overall reconstruction quality and the compression capacity of high-frequency striped patterns, our strategy is to identify high-frequency areas. To do this, we propose adding attention modules to the network.

We are also working on training the network by adding a perceptual cost function to the rate-distortion trade-off. Our perceptual cost function allows us to extract areas during training that were previously blurred by the $L2$ norm optimised compression scheme. This new trade-off is also the subject of work on optimising the network's hyperparameters using an automatic multi-objective balancing strategy. Finally, we propose a quasi-lossless compression method with the addition of specialised compression on the high frequencies of the residual image. Through the use of filtering and thresholding of our residuals, we manage to extract structural information contained in the reconstruction error.

Starting with **Chapter 4**, we consider the extension of our compression method to include other processing within the satellite. In this chapter, we present the state-of-the-art in joint image processing methods, focusing only on demosaicking, denoising and compression, the three processes at the heart of the classical image processing chain. These joint methods offer several advantages over sequential processing, both in terms of reconstruction quality and complexity reduction.

In **Chapter 5** we detail the merging of denoising and demosaicking within our compression network. This joint processing makes it possible to address both compression and reconstruction performance issues and the complexity problem by reducing the number of steps in the processing chain. We begin with a proof of concept of the model, combining raw, and therefore colour filtered, data with a compression network that enforces an output colour representation.

This base model is improved with the addition of a guidance branch during training, the aim being to force a larger intermediate representation of image features. This fits in perfectly with our previous multi-objective training scheme, without increasing the cost of computing on board the satellite. A pre-processing of the raw data before the network is added to improve the reconstruction. To get as close as possible to real data, we consider noisy data and modify the network training to perform image denoising jointly with the other two operations. Finally, an extension of this work towards a constant quality compression method is envisaged within the framework. By studying the raw image statistics at the beginning of the processing pipeline, we can estimate the quality parameters needed to ensure that the compression never falls below a certain quality threshold, thus guaranteeing constant reconstruction quality.

Author's publications

International conferences

Bacchus, P., Fraisse, R., Roumy, A., & Guillemot, C., [2022], Quasi lossless satellite image compression, *IGARSS 2022 - IEEE International Geoscience and Remote Sensing Symposium*, 1532–1535, https://doi.org/10.1109/IGARSS46834.2022.9883135

Bacchus, P., Fraisse, R., Roumy, A., & Guillemot, C., [2023c], Joint compression and demosaicking for satellite images, *ICASSP 2023 - IEEE International Conference on Acoustics, Speech and Signal Processing*, 1–5, https://doi.org/10.1109/ICASSP49357.2023.10096011 **Top 3% of all accepted papers**

Bacchus, P., Fraisse, R., Guillemot, C., & Roumy, A., [2023a], Filtered Residual Compression for Satellite Images, *IGARSS 2023 - International Geoscience and Remote Sensing Symposium*, 1–5, https://hal.science/hal-04125811

National conferences

Bacchus, P., Roumy, A., Fraisse, R., & Guillemot, C., [2023d], Compression quasi sans perte d'images satellites par filtrage de residus, *GRETSI 2023 - XXIXeme Colloque Francophone de Traitement du Signal et des Images*, 1–5, https://hal.science/hal-04156801

Under preparation for publication in a journal

Bacchus, P., Fraisse, R., Guillemot, C., & Roumy, A., [2023b], Full processing pipeline for satellite images: Denoising, Compression, Demosaicking

Chapter 1

Satellite imaging system

Contents

1.1 Overview of the processing pipeline

Airbus is preparing to launch a new constellation of small observation satellites [Lebègue et al., 2020] with increased spatial resolution. Compared with the Pleiades constellation, which has a resolution of 70 cm for the panchromatic band and 280 cm for the multispectral bands [Gleyzes et al., 2012], this constellation will have a geometric resolution of 50 cm for each RGB colour band. The constellation also differs in the nature of its sensors. In fact, the acquisition is not of the push-broom type as for the Pleiades, but of the starer type, similar to the Chinese satellite Jilin-1. The different types of acquisition are shown in Figure 1.1.

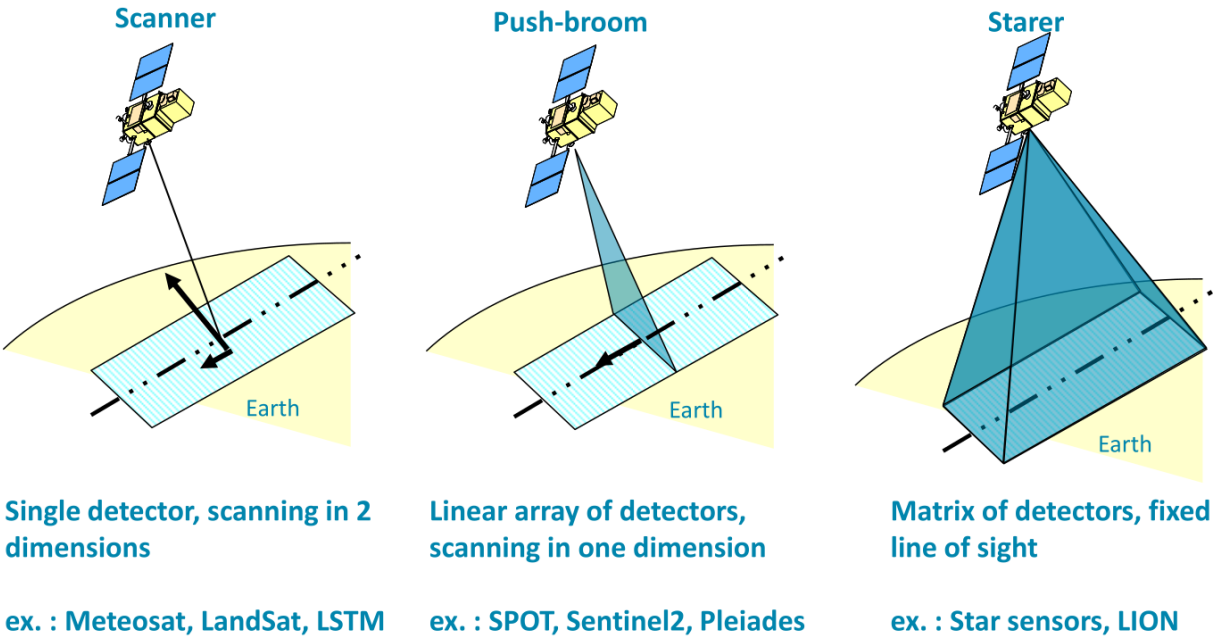


Figure 1.1 – Different types of acquisition of satellite images.

In starer acquisition, a matrix of sensors captures only 3 visible bands together, similar to standard digital cameras. A complete image is captured with no focal plane shift. A colour filter array (here a Bayer filter) determines which colour is captured at each pixel position for the photodiode in the sensor. Demosaicking, the processing required to reconstruct a full colour image from sub-sampled channels, is performed on the ground. With push-broom satellites, the satellite takes bands of an area to reconstruct an image. Each band is acquired separately: red, blue, green, near-infrared and panchromatic (from

400nm to 800nm). As the colour bands are acquired separately, each pixel in each band does not capture exactly the same position on the Earth due to the different wavelengths of the bands and the optical effects that depend on them (e.g. diffraction). To obtain high-resolution colour images, additional processing must be performed to re-arrange the spectral bands.

In this thesis, we consider a simple image processing pipeline for a starer satellite, shown in Figure 1.2. The only processing of the raw data on board the satellite is compression to reduce the bandwidth required for transmission. The amount of data involved is significant: the satellite is estimated to take an average of several thousands images per day per satellite, each scene at 150 Mpix per second. Further processing, such as demosaicking and denoising, is carried out on the ground due to the high level of complexity of the task. Additional processing includes a wide range of corrections and calibrations, from geo-referencing to focal plane repositioning, but is beyond the scope of this thesis. Prior to delivering the data to the end user, some higher level processing such as classification or Digital Surface Model (DEM) can be performed.

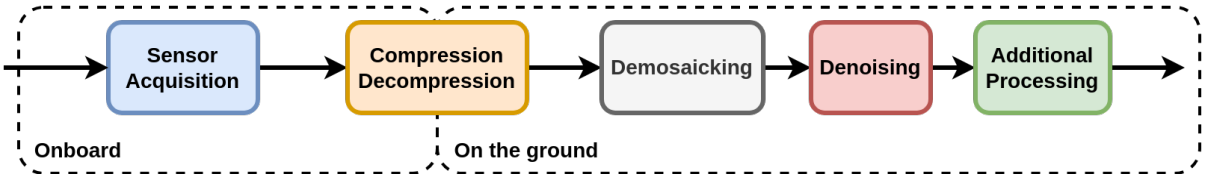


Figure 1.2 – Simplified image processing pipeline for the satellite.

1.2 Image compression

Although lossless compression is possible, it does not reduce the size sufficiently. In this thesis we consider lossy compression, which aims to minimise the bit rate while allowing a certain amount of distortion between the reconstructed image and the original image. The allowed distortion for satellite images is much lower than for conventional images for analysis purposes. Satellite images are highly correlated signals with both spectral and spatial redundancy. Transform coding is used to decorrelate the image so that it can be compressed efficiently. Other methods exist for exploiting redundancy in the image, some of which are specially designed for satellite imagery using the uplink to send information about the temporal redundancy [Aulí-Llinàs et al., 2016]. Quantisation introduces

approximation error into the image before it is entropy encoded into a bitstream. This bitstream is then decoded and the inverse transform is used to recover a reconstruction of the original image.

The Karhunen-Loeve Transform (KLT) [Dony et al., 2001] is used in image processing to decorrelate the data. The Airbus compression baseline uses a modified KLT transform [Ohta et al., 1980]. The RGB channels are transformed into 4 components Y, U, V and W which are compressed independently. For the KLT to work well, the noise must be consistent between the bands. So we need to do an Anscombe transform before doing the KLT to take into account the Poisson distribution of the noise. More details on noise in satellite imagery can be found in Section 1.4. The compression method used is the standard developed by the Consultative Committee for Space Data Systems (CCSDS), which is a wavelet transform similar to the JPEG 2000 codec [Skodras et al., 2001].

1.2.1 Principles of image compression

In the encoder, the quantisation step maps the data to a smaller number of discrete symbols, resulting in some approximation and therefore lossy compression. This quantised representation is losslessly entropy encoded into a bitstream, which is then decoded. Common entropy coding algorithms includes Huffman coding and arithmetic coding. The original data is approximated from the quantised code using inverse quantisation and inverse transform.

Information theory shows that vector quantisation optimises the rate-distortion trade-off and provides the smallest bit rate for a given distortion. However, due to its enormous computational complexity, vector quantisation is rarely implemented and scalar quantisation is preferred. Information theory provides some guarantees for this sub-optimal quantisation. At a high bit rate, uniform scalar quantisation with a variable-length entropy coder is $1.53dB$ lower than the best rate-distortion performance [Wiegand, Schwarz, et al., 2011] with vector quantisation. This holds for any independent and identically distributed data on quadratic distortion metrics.

Despite the good performance of scalar quantisation, it suffers from a major shortcoming compared to vector quantisation, which is the lack of use of correlations in the data.

Images are particularly redundant data, with both spatial redundancy between adjacent pixels and spectral redundancy between channels. If we model quantised pixels in terms of X, a discrete random vector of size n. A property of the joint entropy of X is:

$$H(X) \leq \sum_{i=1}^{n} H(X_i) \tag{1.1}$$

The joint entropy of X is upper-bounded by the sum of the entropies of each X_i. This inequality becomes an equality if X_i are independent. The lowest rate is thus obtained if we can compute the joint entropy, but this proves to be a very complex step that grows exponentially with n. A variable-length entropy encoder that has difficulty exploiting the correlations will use a sub-optimal entropy model, resulting in a higher bit rate. To overcome this problem, we use a transform to reduce the correlation in the data so that the joint entropy is close to the sum of the individual entropies. If we consider Y the transformed quantised pixels with almost independence between Y_i then

$$H(Y) \lesssim \sum_{i=1}^{n} H(Y_i) \tag{1.2}$$

If the inverse transform decorrelates sufficiently the signal, the joint entropy of the quantised code is close to the sum of the entropy of the discrete random variable Y_i. After decoding the bitstream, the inverse transform is applied to the quantised code to recover the approximation of the original image. For this reason, transform coding is the most common type of compression. Hand-crafted transform types include the Discrete Cosine Transform (DCT) used in the JPEG compression standard [Wallace, 1992] and the Discrete Wavelet Transform (DWT) used in the JPEG 2000 standard [Skodras et al., 2001] or the CCSDS standard [CCSDS, 2017].

1.2.2 JPEG 2000 and the wavelet transform

A compression standard defines a set of rules for processing the input data and formatting the behaviour of the bitstream. It provides a degree of interoperability between bitstream senders and receivers. New standards can respond to new needs, such as increased pressure on bandwidth or additional constraints on complexity. Standards are set by companies and expert groups such as the Joint Photographic Experts Group (JPEG). The latter has developed widely used codecs such as JPEG and JPEG 2000.

JPEG 2000 is based on the Discrete Wavelet Transform (DWT) applied to the rows and columns [Skodras et al., 2001]. The DWT decomposes the image into frequency sub-bands, transforming the representation into a pyramidal structure. The low and high filters (both horizontal and vertical) create 4 bands, followed by down-sampling at each iteration, as shown in the example in Figure 1.3:

1. LL: Low filter horizontally and vertically
2. LH: Low filter horizontally and High filter vertically
3. HL: High filter horizontally and Low filter vertically
4. HH: High filter horizontally and vertically

The iterations continue on the low frequency part of the sub-bands (LL). The transform is reversible and the original image can be recovered. This transformed image has a higher energy compression than the DCT [Skodras et al., 2001].

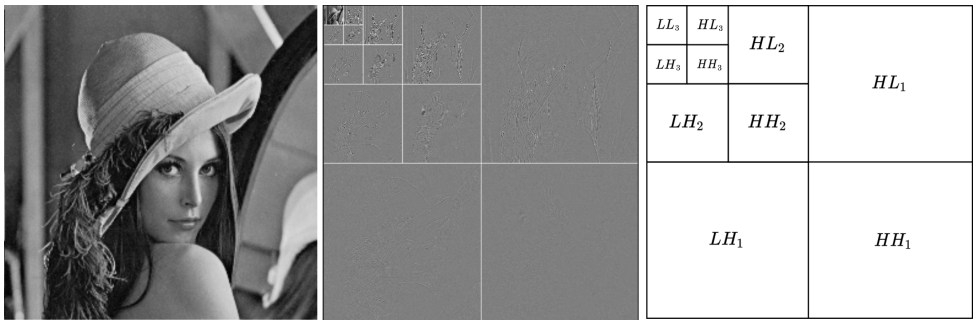


Figure 1.3 – Wavelet transform example. Image of Lena and level 3 DWT decomposition with the sub-band structure.

The full JPEG 200 codec includes additional processing in the compression pipeline. An example is the colour transformation before the DWT on each channel. The YUV colour space with one luminance and two chrominance channels is often preferred for image compression. This is because the human eye is more sensitive to brightness than to colour, so most of the energy is concentrated in one channel. Luminance is essentially the sum of the RGB channel and chrominance is the difference between luminance and the red and blue channels. Chrominance can also be down-sampled to further increase the compression of the input image.

1.2.3 CCSDS 122: a standard for satellite images

The Consultative Committee for Space Data Systems (CCSDS) is made up of the world's major space agencies and defines standards for space data and information systems. These standards must meet the requirements of space missions while allowing interoperability between agencies. For compression, these standards cover a wide range of applications with lossy image compression [CCSDS, 2017] but also hyperspectral predictive image compression and lossless compression [Huang, 2011a].

The CCSDS recommended standard for image compression was first released in 2005 [CCSDS, 2005] with a second version in 2017 [CCSDS, 2017] bringing more backward compatibility and support for higher dynamic range. It offers progressive lossy to lossless compression, quality or bit rate constant compression, and low complexity for high throughput. It has been specifically designed to target high bit rate compression while taking complexity requirements into account.

The codec relies on the DWT, similar to JPEG 2000, followed by a bit-plane encoder that encodes the transformed code. For the DWT it uses a 9/7 bi-orthogonal filter, which offers better rate-distortion performance than the 5/3 filter at the cost of increased complexity. This increase in compression complexity is offset by a limited number of additional features compared to the JPEG 2000 codec while also being specifically implemented for onboard hardware and software. This standard is essentially a modification of JPEG 2000 for the satellite imagery use case.

1.3 Image demosaicking

1.3.1 Colour filter array

The colour filter array is compressed on board using the CCSDS 122.0-B-2 standard [CCSDS, 2017] and transmitted to the ground. Once on the ground, the bitstream must be decompressed and demosaicked to recover a full colour image, as shown in Figure 1.4. The images we use are from starer-type satellites, which use matrices of detectors. In particular, the satellite uses conventional digital sensors.

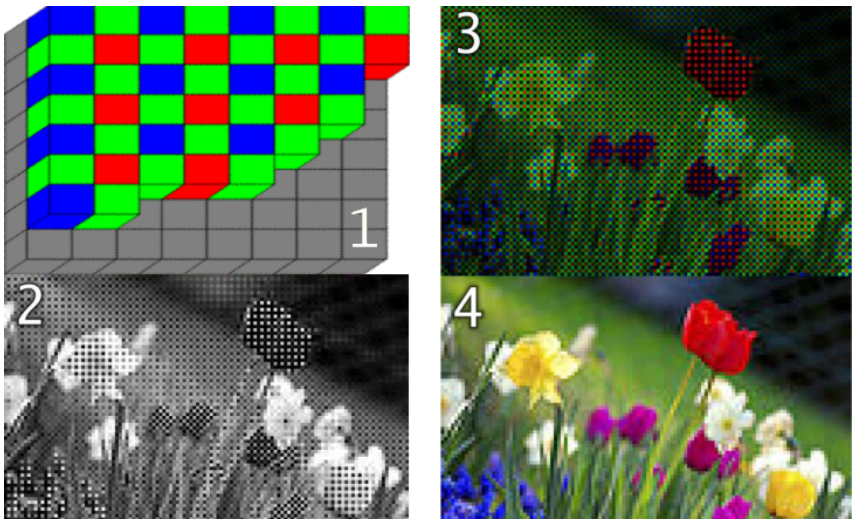


Figure 1.4 – 1) Bayer CFA. 2) Output of a sensor with the Bayer filter. 3) Output colour coded with Bayer filter colours. 4) Reconstructed image after interpolation of missing colour information. Source: Wikimedia Commons

Digital sensors, such as CMOS image sensors, are popular because of their cost and performance. These photodiodes convert light into voltage to measure intensity but only capture monochrome intensity. To overcome this, colour filter arrays (CFAs) are stacked on the photodiodes, the most popular being the Bayer CFA so that each pixel in the raw image is red, green or blue depending on the Bayer profile. The spectral sensitivity of the human eye is highest at red, blue and green wavelengths, with a peak at green. The demosaicking process is used to recover the missing information from each colour channel by interpolation.

Standard demosaicking algorithms include bi-linear interpolation [Losson et al., 2010], gradient corrected interpolation [Buades et al., 2011b; Malvar et al., 2004] or directional filtering with a posteriori decision [Menon et al., 2006]. The method used in the Airbus baseline is a modified version of the Hamilton-Adams algorithm [Buades et al., 2011b] to include some form of denoising.

1.3.2 Hamilton-Adams demosaicking

Demosaicking is similar to a 2x2 super-resolution problem since we interpolate missing information from sub-sampled images. The Hamilton-Adams demosaicking algorithm first interpolates the green channel only and then the other two channels which are initially twice as sub-sampled [Buades et al., 2011b].

The horizontal and vertical gradients at each missing green pixel are computed using the first and second-order partial derivatives using the green and red/blue pixels respectively [Buades et al., 2011b]. Depending on the main direction of the gradient, the horizontal or vertical average of the neighbouring green values is used to estimate the missing value. This is corrected by the second-order derivative of the gradient in that direction. Once the green channel is fully recovered, the red and blue channels are interpolated using bi-linear interpolation on the difference $R - G$ and $B - G$.

1.4 Image denoising

1.4.1 Generalities

The raw image, although noisy, is compressed on board according to the CCSDS 122.0-B-2 standard [CCSDS, 2017] and transmitted to Earth. Once the bitstream is decompressed and demosaicked, denoising is performed. Denoising is a costly process but is necessary because noise is added during acquisition and onboard processing. This noise model affecting satellite imagery is not purely Gaussian and requires some transformation before denoising algorithms can be applied.

Denoising is a widespread research topic. Standard methods include non-local algorithms that rely on similarities in the whole image rather than the context of a pixel's neighbourhood. In the non-local means algorithm, a pixel is replaced by an average of similar pixels across the whole image, rather than just around the pixel [Buades et al., 2011a]. An improved version, known as non-local Bayes, uses patches that are replaced by a weighted average of the most similar existing patches [Lebrun et al., 2013]. BM3D also uses patch averaging, but in a transform domain, and groups similar DCT 2D patches into 3D groups to increase sparsity [Lebrun, 2012]. The denoising method currently used in the Airbus baseline is a non-local Bayes modified to include Hamilton-Adams demosaicking.

1.4.2 Noise model

Noise occurs in several parts in the image processing pipeline. Sensor acquisition introduces shot noise (Poisson distribution) due to the discrete nature of light, which is significant at high light intensities. Photodiodes are affected by thermal noise from electrical charges and the signal is also altered by read noise in the camera electronics. Both noises, known as dark noise, follow a Gaussian distribution, the former being independent of intensity and the latter only relevant to low-intensity signals. For daylight images with our sensor, the model estimates a mean dark noise with a dynamic around 3 compared to the 2^{12} dynamic range and a shot noise of about 15 for shot noise.

The noise model depends on the pixel intensity and takes into account the dark and shot noise. It is given by

$$\sigma = \sqrt{A + B * L} \tag{1.3}$$

Where L is the pixel intensity, A and B are sensor parameters which are around 6 and 0.05 respectively. Standard denoising algorithms such as non-local Bayes [Lebrun et al., 2013] or BM3D [Lebrun, 2012] can be applied to recover a noiseless image.

1.4.3 Variance stabilising transform

State-of-the-art denoising methods assume additive white Gaussian noise [Lebrun et al., 2013; Lebrun, 2012]. To make this assumption hold for the satellite noise model, a variance stabilising transform must be used to transform the Poisson distribution into an approximate Gaussian distribution. The Anscombe transform [Anscombe, 1948] is widely used and is given by:

$$A : x \mapsto \frac{2}{B}\sqrt{B \cdot x + \frac{3 \cdot B^2}{8} + A} \tag{1.4}$$

After standard denoising methods have been applied to the variance transformed image, the inverse Anscombe transform must be performed to recover the original image. The algebraic inverse is given by

$$A^{-1} : x \mapsto \frac{B \cdot x^2}{4} - \frac{3 \cdot B}{8} - \frac{A}{B} \tag{1.5}$$

However, this inverse introduces some bias. Asymptotically unbiased or exactly unbiased inverses exist in the literature [Makitalo & Foi, 2011]. The Anscombe transform is for a pure Poisson distribution, which is not quite the case in our noise model, but since the Gaussian noise is marginal compared to the Poisson noise, we assume that the assumption holds.

1.4.4 Non-local Bayes

Non-local Bayes is an improved version of the non-local means denoising algorithm, which use the weighted mean of the most similar patches in an image to denoise a patch [Buades et al., 2011a]. The non-local Bayes method computes optimal patches in terms of the Bayesian minimum mean square error [Lebrun et al., 2013]. The covariance matrix of a patch is computed to measure the variability of the patch compared to the one we are denoising.

Non-local Bayesian denoising requires two steps. The first step finds the most similar patches in a 3D block, which is filtered and thresholded before an inverse 3D transform. An estimate of the denoised image is computed by aggregating all the other estimates. The second step is essentially the same, except that it uses the result of the first step as an oracle [Lebrun et al., 2013] to improve the search for similar patches.

The denoising performance is important, but the algorithm also suffers from a number of problems. Firstly, the number of hyperparameters is quite high, with 9 to be set between the first and second processing steps [Lebrun et al., 2013]. The other point is the computation time. Despite its high efficiency, especially when compared to BM3D, it's still quite significant when processing a large amount of data. For typical image size and hardware (774x518), the denoising time is in the order of 20s [Lebrun et al., 2013]. For 2000x2000 satellite images, the order of magnitude increases to 10 minutes. This implies a cost that is too high to perform the task in flight and remains a considerable cost on the ground.

1.5 Quality evaluation

The quality of each processing must be evaluated to measure its ability to recover the signal. We focus on two objective measures, PSNR and SNR, which are widely used in the literature to measure the distortion between a reconstructed signal and its ground truth. We also explain the Bjøntegaard metric to compare two rate-distortion curves.

1.5.1 PSNR

The Peak Signal to Noise Ratio (PSNR) is a pixel-based distortion used in many areas of image processing. It measures the accuracy of our image compared to the truth on the ground. Given a reference signal I and a degraded image $\hat{I}$ we have:

$$\begin{aligned} PSNR(I,\hat{I}) &= 10log_{10}(\frac{d^2}{MSE(I,\hat{I})}) \\ &= 20log_{10}(d) - 10log_{10}(MSE(I,\hat{I})) \end{aligned} \tag{1.6}$$

Where d is the bit depth of the image which in our case is $d = 12$. MSE is the mean square error between the two images:

$$MSE(I,\hat{I}) = \frac{1}{nm}\sum_{i=0}^{n-1}\sum_{j=0}^{m-1}(I(i,j) - \hat{I}(i,j))^2 \tag{1.7}$$

Although PSNR is not a direct measure of image quality as perceived by the human visual system and differs from its evaluation, it has become a prevalent standard for evaluating image codec development.

1.5.2 SNR

As an objective metric, we also use the Signal to Noise Ratio (SNR). Given a reference signal I and a degraded image $\hat{I}$, the SNR is calculated as González et al., 2009:

$$SNR(I,\hat{I}) = \sqrt{\frac{\sum_{pix} I[pix]^2}{\sum_{pix}(I[pix] - \hat{I}[pix])^2}} \tag{1.8}$$

SNR is used in the remote sensing community, but not so much in the signal processing and compression community. Satellite images have a dynamic range that can be large (2^{12} in our case), but they most often fill values up to 2^{11} or even 2^{10}, so SNR is often preferred because it is the mean of the signal and not the peak of the dynamic range that is used to compute the metric. It also makes it easier to compare compression noise with instrument noise, since we know from the noise model the amount of noise in the image before compression.

1.5.3 Bjøntegaard metric

Comparing rate-distortion curves is difficult because you need to have exactly the same bit rate or distortion point to compare the results. To overcome this problem, Bjøntegaard proposed to calculate an average difference between two rate-distortion curves [Bjøntegaard, 2001]. The two curves are interpolated with third-degree polynomials and two metrics, defined as the average bit rate or PSNR differences between two codecs, are computed to evaluate the relative performance of one rate-distortion curve over the other.

$$BD - PSNR = \frac{1}{R_h - R_l} \int_{R_h}^{R_l} (C_2(r) - C_1(r))\, dr \tag{1.9}$$

$$BD - Rate = \frac{1}{D_h - D_l} \int_{D_h}^{D_l} (C_2(D) - C_1(D))\, dD \tag{1.10}$$

Where C_1 and C_1 are the two curves and R_h, R_l, D_h, D_l are the lower and upper limits of the domain. The BD rate is expressed in percent and the BD PSNR is expressed in decibels.

1.6 Dataset

The satellite images used in this thesis are simulated images from Airbus Defence & Space new satellite constellation, which will be launched next year. The acquisition is of the starer type with a matrix of sensors with a geometric resolution of 50 cm. These simulated satellite images are based on airborne data acquisition campaigns sub-sampled from to 10 cm to 50 cm resolution. They serve as the RGB ground truth for each test.

The dataset we use consists of 300 12-bit RGB images of size 2000 x 2000 with a 50 cm resolution, covering areas around Lyon, France, provided by Airbus Defence and Space. We use 14 images for the test dataset, the rest is used as training data. Examples of images are shown in Figure 1.5 with images of the city, suburbs, countryside and industrial areas around the city. The dataset does not contain sea images or cloudy images, although these are common types of satellite images. The thesis focuses only on the most challenging images with high entropy.

Figure 1.5 – Simulated 12-bit satellite images around Lyon, France.

In Chapter 5 we will consider raw data estimated from the RGB ground truth. We create 300 pairs $(\mathbf{I}, \mathbf{I_B})$ of RGB noiseless ground truth images and their Bayer CFA counterparts. We use a **GRBG** pattern as the CFA filter with a red pixel in the right position, a blue pixel in the lower left position and green pixels elsewhere in a 2 x 2 square area. We also add noise when considering denoising the raw data and create a pair $(\mathbf{I}, \mathbf{I_{NB}})$ with the noise model

$$\sigma = \sqrt{A + B * L}$$

of the sensor as seen in Section 1.4.2. Where L is the pixel intensity and A and B are the parameters which are around 6 and 0.05 respectively. With the noise captured by the sensor, we get images of around 120 SNR for images that have an average value of 800 over the 2^{12} dynamic range.

Chapter 2

Overview of deep learning models for lossy image compression

Contents

2.1 Introduction

Deep neural networks have become a powerful data-driven tool for solving problems previously tackled with model-based methods. The first major contributions to image compression networks date from 2016, with the first recurrent neural networks [Toderici et al., 2015] dedicated to image compression. In a few years, they have managed to go from the performance of early-century codecs such as JPEG 2000 to the best handcrafted codec, VVC [D. He et al., 2022]. With model-based compression, performance is improved by refining each element of the compression framework. This is a time-consuming effort that requires many manpower to make small improvements to each part of the codec. In network-based compression methods, performance is improved in an end-to-end manner by working together on the architecture of the network and the training (i.e. loss functions, data set, optimisation).

Today, there is a wide variety of networks and numerous contributions to improve each point of the network [Hu et al., 2021]. This chapter provides a comprehensive review of the literature on end-to-end learned compression. It presents various design options and components and outlines the challenges and limitations they face. In particular, we will focus on one architecture that will form the basis of our work: the seminal work of Johannes Ballé on the hyperprior autoencoder [Ballé et al., 2018]. This is the backbone of most of the literature today, and we will analyse this architecture in depth, as well as subsequent improvements. Other types of architectures will be discussed such as recurrent networks and adversarial generative networks.

2.2 Image compression with a scale hyperprior

2.2.1 Autoencoder architecture

Autoencoders are a type of network on which the compression literature relies heavily. They are designed to reconstruct input data by extracting its main features. The training enforces the output to be close to the input, with often a reduction in dimension within the network. In fact, the network can be divided into an encoder and a decoder with a minimum dimension between them, known as the bottleneck. This bottleneck holds the latent representation of the input, which is similar to transform coding in traditional codecs such as JPEG 2000 [Skodras et al., 2001].

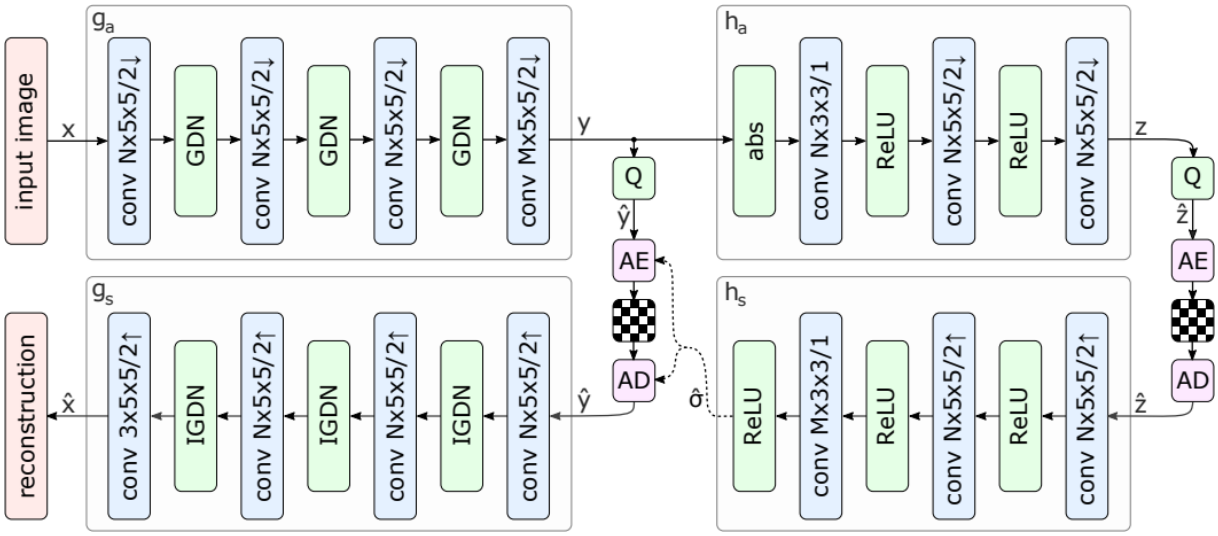


Figure 2.1 – Hyperprior architecture. Source: [Ballé et al., 2018]

The Figure 2.1 shows the complete hyperprior architecture, which is essentially two connected autoencoders. The first one does the dimension reduction of the input data and the second one, the hyperprior, is used to improve the entropy model. The entropy models and the role of the hyperprior will be discussed in more detail later in this section. If we leave aside the hyperprior, we find ourselves with a standard autoencoder, which has been a first step towards deep image compression [Ballé et al., 2017; Theis et al., 2017].

Given an input image x, the encoder acts as a non-linear transform T_E with parameters θ_E to produce the latent representation y which is processed by a quantisation function Q into $\hat{y}$:

$$\hat{y} = Q(T_E(x, \theta_E)) \tag{2.1}$$

The discrete code $\hat{y}$ is entropy encoded and decoded with an entropy model $p_{\hat{y}}(\hat{y})$ with parameters θ_y learned by the network. To recover the image $\hat{x}$ we need the decoder which takes the place of an inverse transform T_D with parameters θ_D so that:

$$\hat{x} = T_D(\hat{y}, \theta_D) = T_D(Q(T_E(x, \theta_E)), \theta_D) \tag{2.2}$$

2.2.2 Layers analysis

The key component of neural networks in image processing is the convolutional layer. The input is spatially convoluted with a filter to produce a value from a local region rather than the classical multi-layer perceptron that fails to take into account the contributions of its spatial neighbours. The input is down-sampled four times through convolutional layers while increasing the number of feature maps from 3 to N in the hidden layers to M at the bottleneck layer. The resulting latent representation is still of lower dimension than the input if the bottleneck layer consists of less than $3 * 4^4 = 768$ feature maps for RGB input. This larger number M of filters before the latent code creates a wide bottleneck. The purpose of this is to achieve comparable performance at low values of N and to handle higher bit rates more efficiently [Ballé et al., 2018].

The activation function used in the forward and inverse transform differs greatly from the standard ones such as Relu or sigmoid. The Generalised divisive normalisation (GDN) is a parametric function introduced by [Ballé et al., 2016a, 2016b] to reduce the redundancies brought by the convolutional layer. This normalisation processes each channel independently and divides each filter output by a measure of the overall filter activity. In the equation (2.3) x_i and y_i refer to the input and output data. α_{ij}, β_i, γ_{ij} and ϵ_i are trainable parameters, i and j the channel index.

$$y_i = \frac{x_i}{(\beta_i + \sum_j \gamma_{ij} |x_j|^{\alpha_{ij}})^{\epsilon_i}} \tag{2.3}$$

The approximate inverse transform required in the decoder is:

$$y'_i = x'_i * (\beta'_i \sum_j \gamma'_{ij} |x'_j|^{\alpha'_{ij}})^{\epsilon'_i}) \tag{2.4}$$

The GDN provides a more accurate estimate of the optimal transform compared to conventional activation functions. These layers are inherently more complex than standard functions but they show sufficient performance gain to be used with great effectiveness in these shallow architectures. Image compression architectures are not as deep compared to many other image processing networks which increases the impact of GDNs.

2.2.3 Loss functions and training

The set of parameters $\theta_E, \theta_D, \theta_y$ correspond to to the weights of the convolutional and GDN layers of the encoder, decoder and entropy model. They are learned over a rate-distortion trade-off to minimise the bit rate used to compress the latent representation while minimising the distortion between the input and reconstructed image. The loss function $\mathcal{J}$ measures this trade-off:

$$\begin{aligned} \mathcal{J}(\theta_E, \theta_D, \theta_y) &= D(x, \hat{x}) + \lambda R(\hat{y}) \\ &= D(x, T_D(Q(T_E(x, \theta_E)), \theta_D)) + \lambda R(\hat{y}) \end{aligned} \tag{2.5}$$

Here λ controls the trade-off between the the distortion measure and the bit rate. The distortion D measures the dissimilarity between the input image and the reconstructed image. Often the $L2$ norm is used:

$$D(x, \hat{x}) = \frac{1}{N} \sum_{n=1}^{N} ||x - \hat{x}||_2 \tag{2.6}$$

The bitstream size generated by entropy coding is lower bounded by the entropy of the discrete probability distribution of the quantised vector $\hat{y}$ [Ballé et al., 2018]. The rate increase is due to the mismatch between the probability model inferred by the coder design and the actual discrete probability distribution. The rate is given by the Shannon cross entropy between the two:

$$R(\hat{y}) = -\mathbb{E}[log_2 p_{\hat{y}}(\hat{y})] \tag{2.7}$$

However, a problem arises during training due to the non-differentiability of the quantisation process. Since the derivative is zero or undefined at an integer value, the learning of the network through the backpropagation of the gradient is not possible. To get around this problem the quantisation is replaced by different relaxation methods during training. The work of [Theis et al., 2017] proposes to ignore the quantisation and choose an identity function during training. In the Ballé autoencoder, the quantisation is approximated with an additive uniform noise [Ballé et al., 2017] that simulates the bin size of uniform

scalar quantisation. This modification allows the differential entropy of $\tilde{y}$ to be used as an approximation of the entropy of $\hat{y}$. The differential entropy is also used during training to compute an estimated rate R, which used in the rate-distortion optimisation problem. This process occurs only during training, and quantisation is performed at inference time.

2.2.4 Entropy model

The probability model $p_{\hat{y}}(\hat{y})$ used to entropy encode the latent code $\hat{y}$ must be close to the actual probability distribution to reduce the bit rate of the bitstream.

2.2.4.1 Factorised model

In the standard fully factorised model all the channels of the latent representation are assumed to be independent and identically distributed. The network learns a factorised distribution, which is essentially a differentiable histogram for each channel of the quantised code. Similar to the bit rate estimation, the differential entropy $\tilde{y}$ is used [Ballé et al., 2017]:

$$p_{\tilde{y}|\theta_y}(\tilde{y}|\theta_y) = \prod_{i=0}^{M} p_{\tilde{y}_i|\theta_y^i}(\tilde{y}_i) \tag{2.8}$$

The factorised model θ_y^i learns to model the corresponding $\tilde{y}_i$ for each channel i. This distribution is then used as the entropy model during arithmetic coding.

2.2.4.2 Hyperprior model

To obtain a better entropy model and thus a higher rate-distortion trade-off, the factorised model is replaced by a more powerful entropy model: the hyperprior model [Ballé et al., 2018]. It introduces side information z derived from an additional autoencoder that takes y as input. The same factorised distribution is used to encode z but z is now decoded to produce either the scale of a Gaussian distribution [Ballé et al., 2018] or the mean and scale [Minnen et al., 2018]. The conditional distribution is now given by:

$$p_{\tilde{y}|\tilde{z}}(\tilde{y}|\tilde{z}, \theta_h) = \prod_{i=0}^{N} (\mathcal{N}(\mu_i,\, \sigma_i^2) * \mathcal{U}(-0.5, 0.5))(\tilde{y}_i) \tag{2.9}$$

All parameters of the hyperprior learned by the network are encapsulated in θ_h. This includes the hyperprior encoder, decoder and factorised entropy model. This side information has to be entropy coded, which increases the overall file size, but the bit rate savings due to a better entropy model for $\hat{y}$ far outweigh this drawback. In fact, this entropy model is now image-dependent, resulting in improved performance. The hyperprior architecture achieves similar rate-distortion performance as the HEVC codec [Ballé et al., 2018].

2.3 Improving autoencoder networks

Alongside the hyperprior architecture, other models have been proposed based on convolutional networks. Dumas proposed an autoencoder combined with dictionary learning and sparse representation of the latent code [Dumas et al., 2017] to perform image compression. Different feature extraction convolution layers in the encoder and decoder were tried such as a pyramidal decomposition of the input image to exploit data shared across various levels of analysis [Rippel & Bourdev, 2017]. Cheng used a principal components analysis on the latent representation to obtain a more energy-compact latent code [Z. Cheng et al., 2018]. This contribution was extended contribution with energy compaction-based loss function [Z. Cheng et al., 2019]. Other studies have focused on loss functions such as adversarial loss in the training similar to the fashion of GANs [H. Liu et al., 2018].

However, most contributions to autoencoders are now based on the hyperprior architecture, which has become a new standard for autoencoder networks. We examine a number of contributions aimed at improving the entropy model, the architecture of the network, or its complexity.

2.3.1 Context model

The entropy model of the hyperprior architecture is further improved by including a spatial auto-regressive component which operates on the latent representation rather than the pixel. The error model uses masked convolutions as shown in Figure 2.2 to condition the entropy parameter on the already decoded causal context [Minnen et al., 2018]. This idea has been explored in similar work [Klopp et al., 2018; J. Lee et al., 2019] to reduce spatial redundancies among latent representations.

The limitation is that although encoding is fast because mask convolution can be applied in parallel to all locations, decoding becomes a serial operation because arithmetic decoding and convolution-based prediction are intertwined. We lose the highly parallelisable architecture that could take full advantage of GPUs.

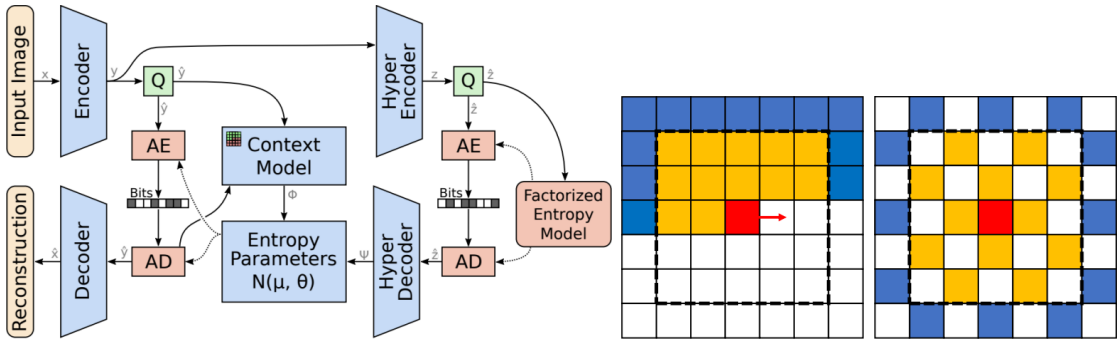


Figure 2.2 – Left: Hyperprior with a context model. Middle: Serial masked convolutional. Right: Checkerboard masked convolution. Source: [D. He et al., 2021; Minnen et al., 2018]

A parallelisable checkerboard context model, shown on the right in Figure 2.2, is used to overcome this limitation [D. He et al., 2021]. It allows a parallel implementation while maintaining the performance of the original masked convolution. This is achieved by reorganising the coefficients and using a double pass for decoding.

2.3.2 Channel-wise entropy model

In order to further improve the hyperprior model and drop the spatial error model with its inherent sequential decoding scheme, a channel conditional entropy model was explored [Minnen & Singh, 2020].

In a standard hyperprior-based model, the entropy parameters used to encode the latent representation y are conditioned on a hyperprior z. In the channel-wise entropy model in Figure 2.3, y is split in slices along the channel dimension. The first slice is decoded based on the hyperprior but other slices use the hyperprior and previously decoded values. This provides more information and thus a more accurate entropy model which leads to a better compression rate.

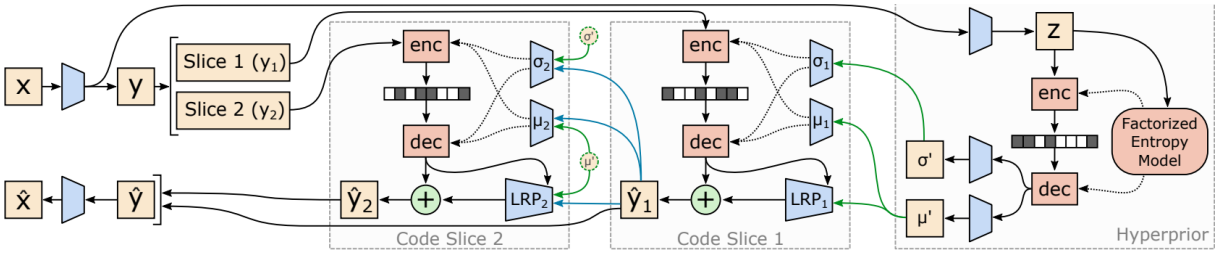


Figure 2.3 – Hyperprior architecture. Source: [Minnen & Singh, 2020]

This idea was simultaneously explored in a coarse to fine hyperprior model [Hu et al., 2020], which also uses hierarchical layers of hyperpriors to reduce spatial redundancy in the data and improve rate-distortion performance. This idea has been refined with a non-uniform grouping scheme to allow a finer subdivision of the slices [D. He et al., 2022]. The first slices use fewer channels in the hyperprior and as the model progresses it becomes more refined with a greater number of channels. All these models allow an adaptive compression that can be more or less refined depending on the number of slices.

2.3.3 Attention modules

Attention modules [Woo et al., 2018] quickly found their way into all kinds of computer vision neural networks, even for compression. Attention layers are used to guide the neural network to the most relevant information within its structure. The basic structure is shown in Figure 2.4. The input passes through a convolutional layer, followed by a standard multi-layer perceptron and an activation function. The result is fed into a channel-wise multiplier that also takes into account the input features.

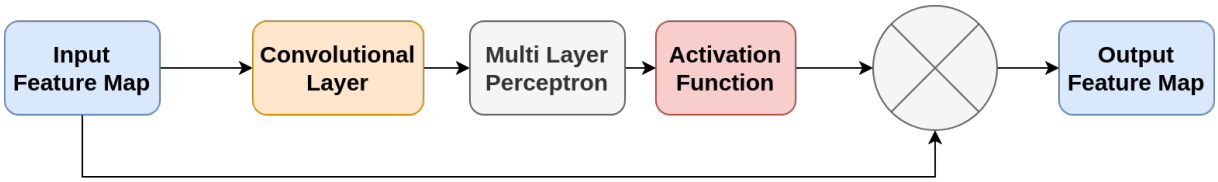


Figure 2.4 – Description of a basic attention module block.

In terms of image compression, this layer assists the network in highlighting the complicated regions of the image to achieve a better trade-off in bit rate distribution to the

demanding parts of the image. It was first introduced in the work of [J. Liu et al., 2020] and refined with more complex attention blocks in [Z. Cheng et al., 2020; Zhou et al., 2019].

2.3.4 Adapting entropy models

In all previous examples, the hyperprior was used to model Gaussian parameters for the entropy model. Entropy coding can be improved by developing a more comprehensive model as seen in [Ladune et al., 2020] which enhance the hyperprior model with additional binary values and an integer. Also, depending on the statistics of the dataset, the distribution of the latent representation may differ from the assumption of a Gaussian distribution. For hyperspectral images, a Student's t-distribution is used to model $p_{\hat{y}}(\hat{y})$ [Y. Guo et al., 2021]. This distribution is used to better capture the anisotropic nature of hyperspectral imagery. Compared to the equation (2.8), the conditional distribution is now given by:

$$p_{\tilde{y}|\tilde{z}}(\tilde{y}|\tilde{z},\theta_h) = \prod_{i=0}^{N}(\mathcal{T}(\tilde{y}_i|0,\nu_i) * \mathcal{U}(-0.5,0.5))(\tilde{y}_i) \tag{2.10}$$

The parameter learned by the hyperprior for a zero-mean Student's t-distribution is the degree of freedom.

This modification of the distribution has also been studied for panchromatic satellite images [Alves de Oliveira et al., 2020, 2021]. The authors propose a simplified version of the hyperprior in Figure 2.5 to model the latent representation with a Laplace distribution.

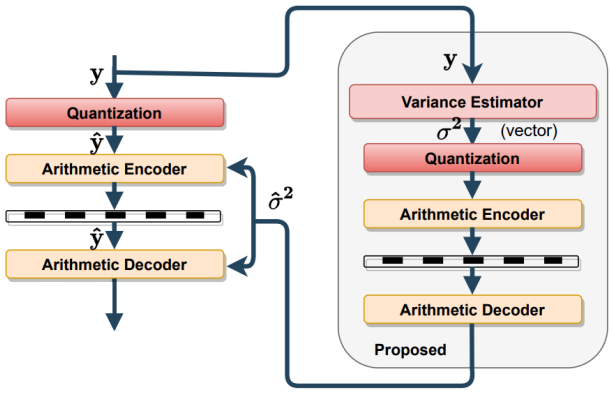


Figure 2.5 – Simplified version of the hyperprior to model a Laplace distribution. Source: [Alves de Oliveira et al., 2021]

The hyperprior becomes a variance estimator whose results are transmitted as side information for a better inference of a zero-mean Laplacian distribution.

$$y_i \sim Laplace(0, b_i) b_i = \sqrt{\frac{Var(y_i)}{2}} \tag{2.11}$$

This illustrates the adaptability of the hyperprior to change the target distribution and adapt to different statistics.

2.4 Alternative network architecture

Only one type of neural network has been introduced so far, autoencoder networks, which are a subset of the classic feed-forward networks. They are the most common in the literature [Hu et al., 2021] due to their effectiveness over a wide range of bit rates, but other architectures have been explored. We want to give a brief overview of some other networks and how they fit into the deep image compression literature. Some were developed in parallel with autoencoders in the early days of deep learning for image compression, such as recurrent neural networks [Toderici et al., 2015]. Some are specifically targeting low bit rates with highly generative reconstruction [Agustsson et al., 2019].

2.4.1 Recurrent neural networks

Recurrent neural networks (RNNs) are a type of neural network that can be applied iteratively to a given sequence of inputs, as shown in Figure 2.6. They allow efficient processing of sequences and are widely used in natural language processing networks. Each output is added as an additional input variable to the next recurrent iteration in a cyclic graph representation [Goodfellow et al., 2016]. However, this arbitrarily long sequence of iterations is the drawback of these RNNs since the network must run sequentially for the next iteration to have an input. Also, these networks are extremely susceptible to vanishing gradients because the layers are only traversed once in each iteration.

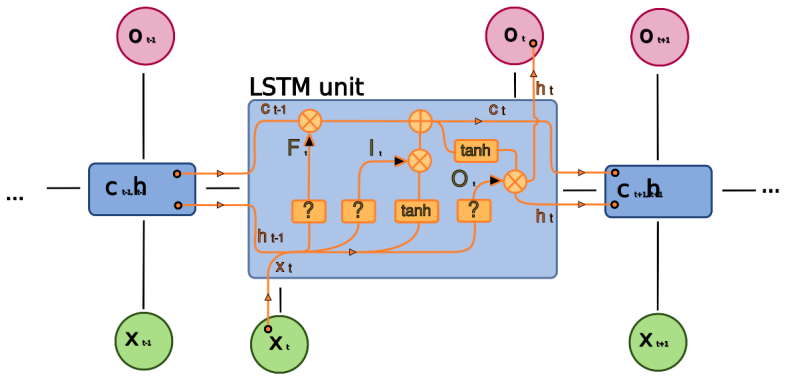


Figure 2.6 – Recurrent neural network with a long short term memory unit. Source: Wikimedia Commons

To overcome this problem, long-short term memory layers (LSTMs) [Hochreiter & Schmidhuber, 1997] are used, as shown in the middle of Figure 2.6. They contain an input gate, an output gate and, most importantly, a forget gate. For each iteration, the LSTM determines which elements from the past to "forget" and which to keep in order to update its hidden state. This is a useful way of avoiding the vanishing gradient problem and ensuring that long-term connections are still considered by the method.

The RNN architecture has been studied in the same period as the autoencoders to take into account a progressive encoding and decoding of the images [Johnston et al., 2018; Toderici et al., 2015, 2017]. They consist of an encoder, a binariser and a decoder containing LSTM units. Similar to how autoencoders work, the encoder transforms the input into a latent representation. The decoder then produces an approximation of the original input image using the received binary code. This method is repeated with the subsequent iteration using the recurrent elements transmitted [Toderici et al., 2015, 2017]. A variable bit rate is naturally obtained with the recurrent architecture, since at each iteration the reconstruction quality increases with the total number of bits.

However, the lower rate-distortion performance, the more difficult training with back-propagation of the gradient over time and the longer inference time to encode an image do not compensate for the advantage of variable compression, which can be achieved by other means and more easily by other architectures. This type of network architecture is less competitive than autoencoders for image compression.

2.4.2 Generative adversarial networks

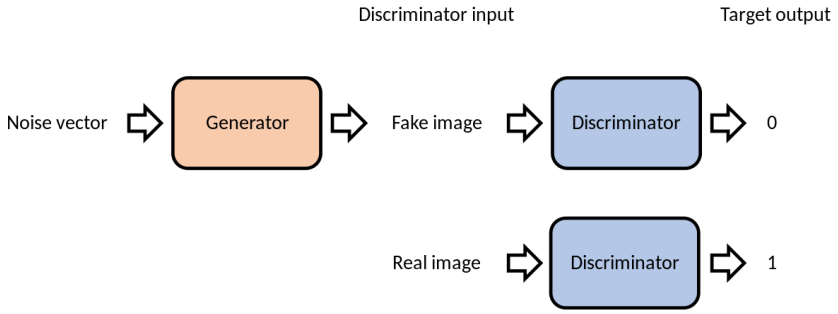


Figure 2.7 – Generative adversarial network. Source: Wikimedia Commons

Generative adversarial networks were first introduced as specialised networks for reproducing statistics similar to the training dataset [Goodfellow et al., 2014]. They are used to generate artificial images that look real and are more concerned with perceptual losses than pure distortion metrics. As shown in Figure 2.7, the idea is that two neural networks are competing in a zero-sum game.

The generator $\mathcal{G}$ outputs content that is as realistic as possible. Its main task is to fool the other network (the discriminator) into believing that the output is not artificial. The discriminator $\mathcal{D}$ tries to determine whether the input given to it is real or fake data. The two networks compete against each other, and in this joint training, both networks gain increased performance in their respective tasks. The focus is less on the absolute quality of an image compared to ground truth, and more on producing convincing images.

These conditional generative adversarial networks are used in image compression to generate high-quality perceptual images from a latent representation of an encoder. They exploit the strengths of GANs to produce extremely low bit rate images [Agustsson et al., 2019] that still have good enough visual quality. Some work has even combined autoencoders in the encoder to produce high-resolution natural images at half the bit rate of the best codec [Mentzer et al., 2020].

However, it is important to note that the emphasis is on perceptual quality, even though the image may look real and with high resolution. This means that the discrepancy between the reconstructed and ground truth images can be high in terms of pixel-based dis-

tortion. These methods can typically produce hallucinations and information not present in the original images and are therefore not suitable for sensitive content such as satellite imagery where high fidelity is critical.

2.5 Conclusion

In this chapter, we have reviewed the literature on deep learning for image compression. A variety of networks are available to meet specific needs, from low bit rate generative compression to recurrent networks. We have focused on the autoencoder network, a good all-around architecture whose performance makes it the state-of-the-art in deep image compression. This literature will form the basis of Chapter 3, where we aim at an end-to-end network optimised for satellite imagery. More specifically, for the work that follows, we consider the architecture of the hyperprior autoencoder [Ballé et al., 2018], which combines very good performance, rivalling standard HEVC, with less complexity than the best existing networks.

Chapter 3

Satellite image compression

This chapter is adapted from [Bacchus et al., 2022, 2023d, 2023a].

Contents

3.1 Introduction

The aim of this thesis is to explore the new deep learning methods used in image compression for the specific needs of satellite images. We seek to obtain a compression network for satellite images, i.e. to obtain the best rate-distortion trade-off taking into account the characteristics of satellite images:

— The use case:
  1. Quasi-lossless compression to preserve information for analysis on the ground
  2. Meet hardware constraints by reducing complexity and memory consumption

— The statistical characteristics of the image:
  1. Pixel-sized details that lead to high entropy images
  2. High dynamic range (12 bits)
  3. Specific scenery

Some of the characteristics presented here are directly related to the hardware constraints of satellites, and thus the need to think not only about the rate-distortion trade-off, as is usually the case for compression networks on natural images, but also about the compromise with reduced complexity. This raises the need to think about multi-use networks and the complexity and size of the networks. However, this is a tricky issue because satellite imagery requires a very high quality of reconstruction for accurate interpretation of the remote sensing data. In fact, we can call this almost lossless compression since we target a distortion much lower than the sensor noise. To achieve this rate-distortion performance, the capacity of the network and therefore the overall complexity must be important. We need to analyse each part of the architecture to understand their respective contributions and propose new schemes accordingly to meet our needs. Finally, this type of data is very different from natural images, in particular it has a high resolution, a wide dynamic range and many high frequency details. This results in high entropy images harder to compress.

From this set of specifications, we can define three main characteristics that satellite image compression algorithms must meet. First, (i) high bit rate compression, tending towards quasi-lossless compression, is required to allow accurate interpretation on the ground. Second, (ii) the algorithms must be of low computational complexity. Third, (iii) satellite images differ from natural images in that they contain pixel-sized details and

have high entropy. So compression must preserve these high-frequency details, which are often removed by the saturating effect of deep learning compression. While the literature on satellite image compression mostly focuses on multi and hyperspectral images [Huang, 2011b; Z. Wang et al., 2021; Yu et al., 2009], we propose a single image compression algorithm based on autoencoders that preserves the three required properties. This type of neural network has recently led to major advances in the field of image compression, with some applications in the field of non-hyperspectral satellite imagery [Alves de Oliveira et al., 2020, 2021], where the focus is on complexity reduction.

In this chapter, we propose new variational autoencoders schemes for satellite image compression to achieve all of the objectives (i), (ii) and (iii). We present our solutions for designing a compression network that addresses the issues raised by satellite imagery. The starting point is the reference image compression network of Johannes Ballé [Ballé et al., 2018]. This architecture focuses on natural images and tends to saturate at high bit rate. Therefore the challenge is to propose solutions to achieve high reconstruction quality for satellite images. We also investigate the number of filters and kernel sizes to determine a trade-off between network complexity and rate-distortion performance. Contrary to previous learning methods [Ballé et al., 2018; Z. Cheng et al., 2020; Minnen et al., 2018; Minnen & Singh, 2020], where several models need to be learned to cover different bit rate, we propose a single network that can operate successfully over a wide bit range thanks to gain units prior to quantisation. We add extra layers and data augmentation to minimise the network saturation effect, which removes high-frequency features.

During training, this takes the form of attention modules and shear mapping to better highlight problematic parts of the data (e.g. striped patterns). We then focus on increasing the rate-distortion performance of our compression network. To do this, we shift from the traditional rate-distortion trade-off to a rate-distortion-perception trade-off by incorporating perceptual loss. A multi-objective method is used to further improve the results by optimising hyperparameters in the loss function. Finally, we investigate a multi-stage network-based extension for quasi-lossless compression. A generic compression network is coupled to a specialised network focused on the compression of the residual image. This residual image is processed so that only significant error patches are compressed, rather than the compression noise. It achieves a low reconstruction error for each type of information contained in the data.

3.1.1 Learned image compression architecture backbone

We chose the hyperprior autoencoder [Ballé et al., 2018] from the literature as our starting point. The reason is both performance and practical. This network achieves performance on par with some of the best traditional codecs such as HEVC [Sullivan et al., 2012] while being relatively simple and compact. The improved versions of this network [Z. Cheng et al., 2020; Minnen et al., 2018; Minnen & Singh, 2020] suffer significantly from a lower parallelism capability, which makes the use of GPUs less efficient and increases the inference time. We will briefly describe the hyperprior autoencoder [Ballé et al., 2018] and more details can be found in Chapter 2 about this network or the literature on deep learning for image compression.

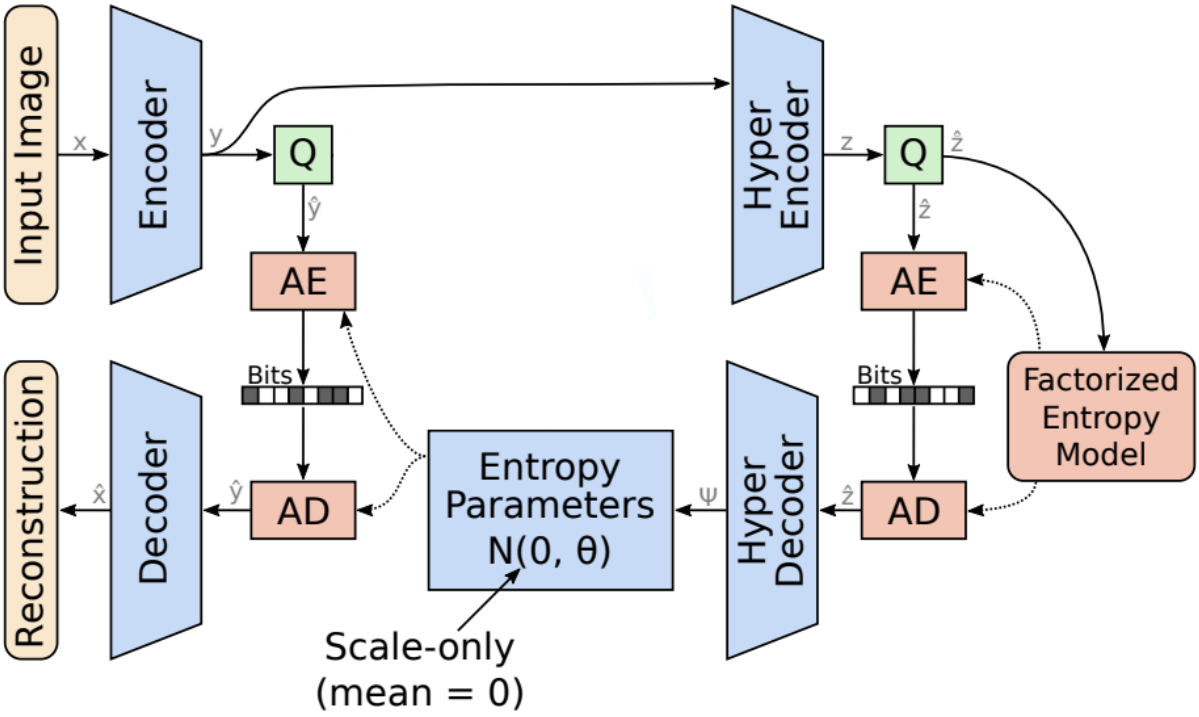


Figure 3.1 – Hyperprior architecture [Ballé et al., 2018]. Source: [Minnen et al., 2018].

The hyperprior architecture [Ballé et al., 2018] consists of two autoencoder networks, as shown in Figure 3.1. The first autoencoder receives the original image x and generates a latent representation y. Quantisation is performed to produce a latent code $\hat{y}$ which is entropy coded into a bitstream. Once decoded, the inverse transform reconstruct $\hat{x}$ from the latent code. The purpose of the other autoencoder (the hyperprior) is to extract

the parameters of the latent representation distribution to improve the entropy model. This entropy model is shared between the encoder and decoder and is used to encode the quantised latent representation into a bitstream. This allows entropy coding models to be adapted to the characteristics of a specific image, as the entropy parameters are estimated for each image.

Each part of the autoencoder is composed of 3 convolutional layers, each with N filters of stride 2 and a 5 x 5 kernel support, followed by a generalised divisive normalisation (GDN) layer. GDN has been shown to be highly efficient in transforming the local joint statistics of images into Gaussians [Ballé et al., 2017].

$$z_i = \frac{x_i}{(\beta_i + \sum_j \gamma_{ij} |x_j|^{\alpha_{ij}})^{\epsilon_i}} \tag{3.1}$$

α_{ij}, β_i, γ_{ij} and ϵ_i represent trainable parameters. i and j are the channel index.

These blocks are linked to the latent representation by a final convolutional layer with M filters: the bottleneck layer. The hyperprior autoencoder follows the same overall layout, except that GDNs are replaced by ReLUs and the bottleneck layer is removed.

All training parameters are learned with the following optimisation problem: a trade-off between a distortion $D(x, \hat{x})$ between the original image x and the reconstructed image $\hat{x}$ and the rate $R(\hat{y})$ of the generated bitstream.

$$J = D(x, \hat{x}) + \lambda R(\hat{y}) \tag{3.2}$$

Where λ is a factor to balance each term of the equation (3.2), and the distortion is chosen as the MSE. This equation, representing the loss function, is then minimised by back-propagation. In the context of compression, the derivative of the quantisation function is either zero or undefined. To overcome the problem of non-differentiability, the quantisation is replaced by uniform noise during training.

3.1.2 Training details

All the experiments in this thesis were carried out using the Python language with the Pytorch [Paszke et al., 2019] machine learning library and the CompressAI [Bégaint et al., 2020] library, which provide tools for developing end-to-end image compression neural network.

The dataset we use consists of 300 12-bit RGB images of size 2000 x 2000 with a geometric resolution (effective ground distance between two pixels) of 50 cm, covering areas around Lyon, France, and was provided by Airbus Defence and Space. We use 14 images for the test dataset, the rest is used as training data. These 286 images are then divided into 4576 non-overlapping 500 x 500 patches, which form our training dataset. The batch size (i.e. the number of training samples that pass through the network at each iteration before gradient descent) is set to 8, and at each epoch the batch size images are randomly cropped to a size of 256 x 256. Some data augmentation is performed to increase the variability of the training data set. This variability increases the representative and exhaustive nature of the training data set. In the long run, this data augmentation will lead to better performing, more reliable models that are more robust to variation. We perform a random amount of $\frac{\pi}{2}$ rotation to induce some rotational invariance in the data.

Without further precision, all the autoencoder parameters presented in this chapter are trained on a rate-distortion trade-off optimisation problem that includes the bit rate $R(\hat{y}, \hat{z})$ of the 2 bitstreams and the distortion $D(x, \hat{x})$ between the original image x and the reconstructed image $\hat{x}$.

$$\mathcal{L}(x, \hat{x}, \hat{y}) = D(x, \hat{x}) + \lambda R(\hat{y}, \hat{z}) \tag{3.3}$$

The hyperparameter λ, which balances the rate-distortion trade-off, is set to values in the interval [0.5,0.8] to target a bit rate of around 2 bits per pixel to achieve the desired quality output. The distortion D chosen to account for image quality is the pixel-based mean square error (MSE) metric. We optimise our approach using the Adam algorithm [Kingma & Ba, 2014] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We use an initial learning rate of $1e^{-4}$, which is halved when the evaluation loss reaches a plateau of 10 epochs.

The metrics used for all further plots are PSNR and SNR, and more details can be found in Section 1.5. The two metrics are used because the two communities (i.e. remote sensing and compression) have their own preferences. SNR is used in the remote sensing community since satellite images have a dynamic range that can be large but most often not fill completely, it implies that the mean of the signal and not the peak of the dynamic range is used to compute the metric. We will add the PSNR metric to our graphs to better compare the results with the state of the art in compression. We decided not to use some of the standard metrics commonly found in the compression literature, such as SSIM, LPC-SI [Hassen et al., 2013], or LPIPS [R. Zhang et al., 2018], because we found that these criteria are not sensitive enough due to the high quality we are aiming for the images. To measure the average gain in SNR or bit rate between two rate-distortion curves, we use the Bjøntegaard metric [Bjøntegaard, 2001]. All the proposed compression networks are compared with JPEG 2000 instead of CCSDS 122.0-B [CCSDS, 2017], as the latter is a slightly degraded version of the former in terms of rate-distortion performance.

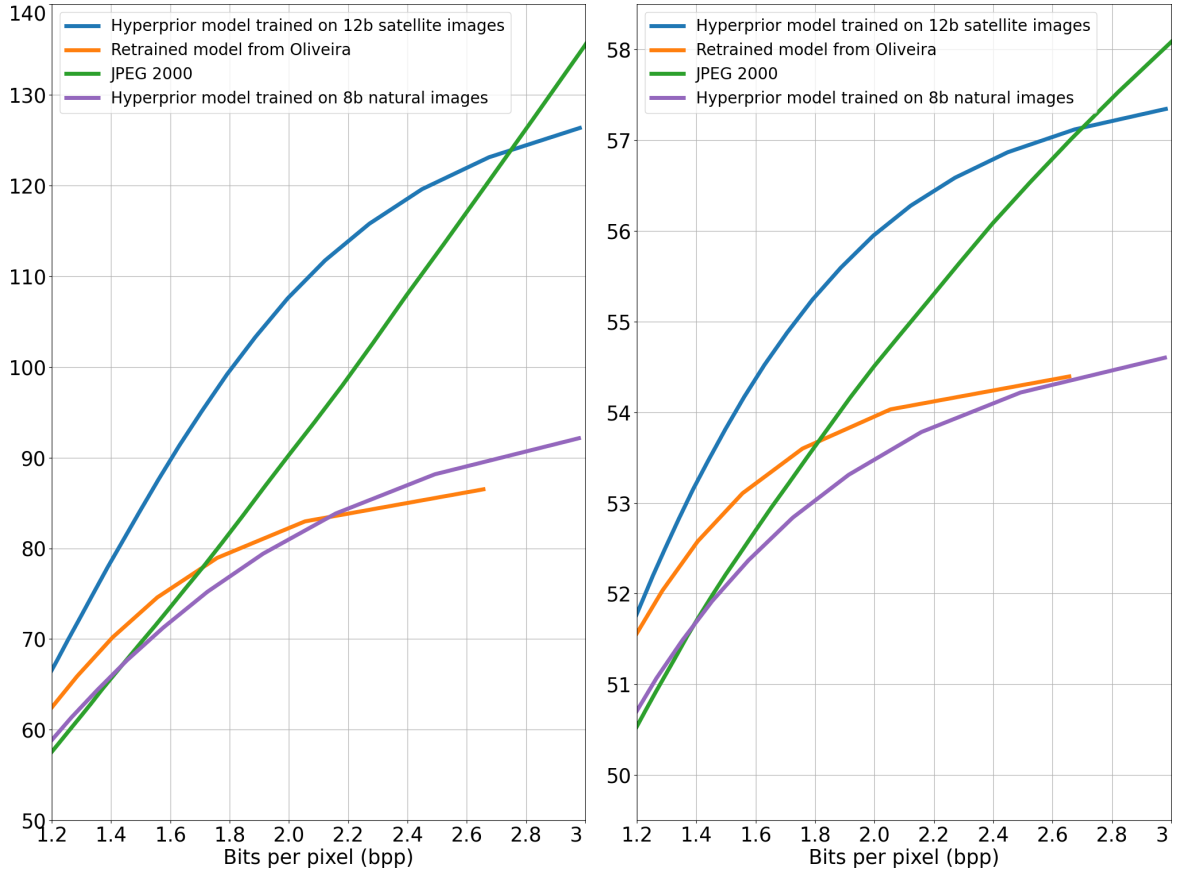
Experiments are run on a variety of GPU hardware, NVIDIA QUADRO RTX 8000, A40 and A100. The training time varies depending on the GPU used but is generally around 8 to 12 hours for 200 epochs. It is faster to train this satellite image compression model than a natural image compression model because our dataset is more specific and the scenery is not as diverse as in natural images. We need fewer training examples and thus reduce the amount of training needed to achieve convergence of the compression model. The inference time (without model loading) is about 1s to encode a 2000x2000 image and 1.5s to decode it. Each model weighs approximately 100MB.

3.1.3 Relevance of the method to a specific dataset

Neural networks have demonstrated their high performance in compression, reaching the level of the best handcrafted codecs HEVC [Sullivan et al., 2012] with hyperprior architecture [Ballé et al., 2018] and VVC [Bross et al., 2021] with more advanced models [Z. Cheng et al., 2020] in just a few years. They are data-driven algorithms, so they can extract the information that defines a particular scene. The compression models described in Chapter 2 were designed for natural images, but training with only satellite images could yield a great improvement as the network weights adapt to this type of image.

The difference between the baseline model trained on natural images and the one trained on satellite data in Figure 3.2 is: BD-SNR = 22.4, BD-RATE = -26.3%. Thus training with satellite images alone reduces the bit rate by 26%. This increases our confidence in the validity of using these learned end-to-end compression methods for our compression problem.

We also compare with other work in the non-hyperspectral satellite image compression literature [Alves de Oliveira et al., 2020, 2021]. However, their dataset differs from ours in that they used panchromatic data with a geometric resolution of 70 cm instead of RGB images of 50 cm resolution. Therefore, for a comparison, we retrain their architecture on our training dataset. This leads to different results as reported in Figure 3.2. Their network is optimised for less complex images, so the architecture has less capacity than ours and fall behind in term of rate-distortion performance at the targeted bit rate.



(a) Distortion measure: SNR (b) Distortion measure: PSNR

Figure 3.2 – Hyperprior models trained on different datasets.

3.2 Designing the architecture to suit our needs

The design of an algorithm must take into account the specific characteristics of the hardware on which it is most likely to operate. We are designing an algorithm for use on a new generation of Earth observation satellites, which have a highly constrained hardware environment. There are two types of constraints: memory footprint of the network and inference time to get a compressed representation.

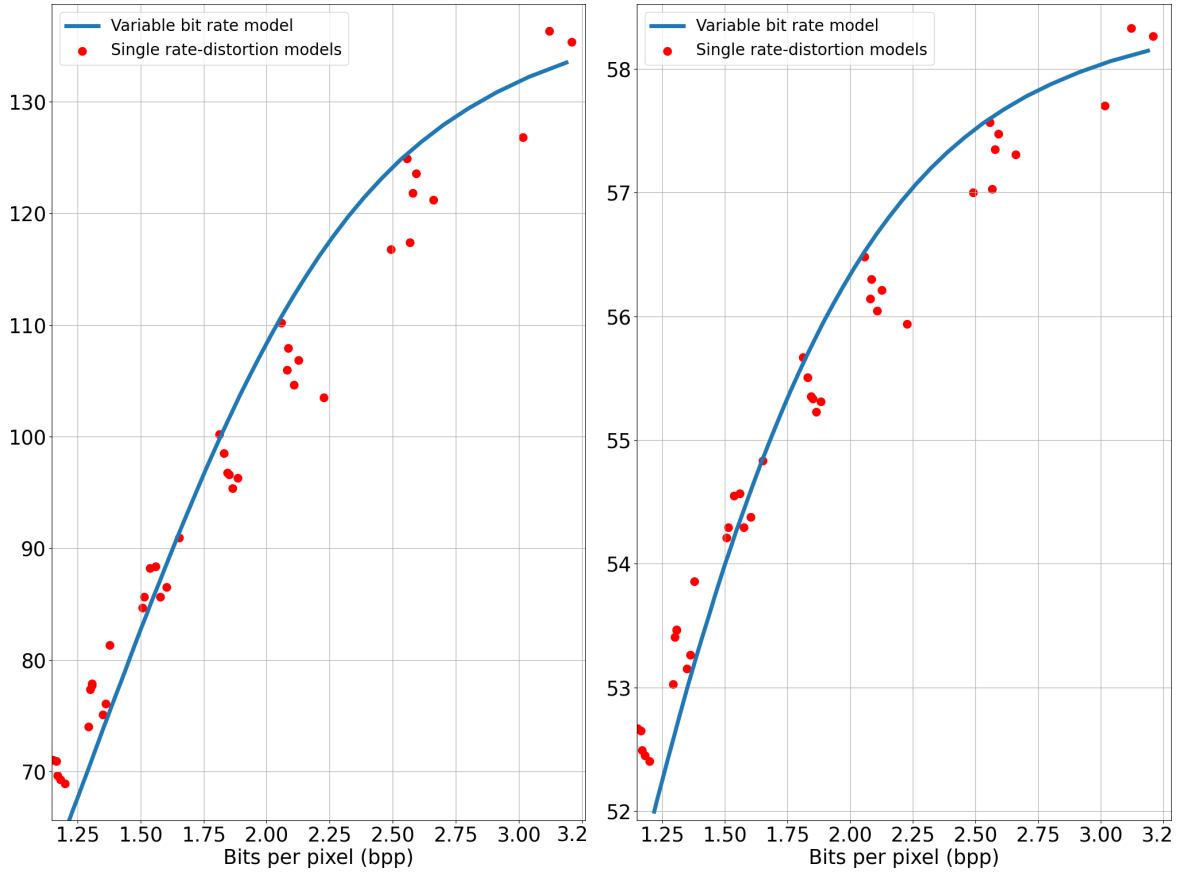
These considerations are the same as for other embedded systems to which the proposed compression method could be applied. We analyse them from the perspective of architecture optimisation to reduce their impact. However, the goal of any compression method is to achieve the best performance for the rate-distortion trade-off. To account for the severe computational constraints inherent in the onboard hardware, a trade-off between performance and complexity must also be considered. The results can lead to a marginal increase in complexity if the performance gain is large enough.

3.2.1 Variable bit rate for multi-usage network

State-of-the-art end-to-end autoencoder compression models [Ballé et al., 2018; Z. Cheng et al., 2020; Minnen et al., 2018; Minnen & Singh, 2020] are trained for a specific rate-distortion point, which requires training and loading multiple models on board. Not only does this require more memory to be able to use the different models on the fly, but it also increases the cost of loading time. There is also the problem of training these multiple networks, which reduces the versatility of the system in the event of a change. To respond to this demand for efficiency, we need one compression model that works over a wide range of bit rates, so that no time or memory is wasted processing multiple models to compress at different rate-distortion targets.

Methods for transforming single rate-distortion point models into multi-bit rate range models mostly revolve around the implementation of gain units [Chen & Ma, 2020; Cui et al., 2021; T. Guo et al., 2020]. They are added at the end of the encoder, just before quantisation, and at the beginning of the decoder. They are essentially scaling factors of the feature maps before quantisation. This has the similar effect of changing the quantization step. The rate-distortion performance is on par with the regular single-point compression model, and only slightly lower overall [Dumas et al., 2018].

Gain unit is an ideal solution to our problem as they require no additional inference time for a large bit rate range usage. We simplify it by not distinguishing between the feature maps and applying the same scaling everywhere. We then train our network while varying the quality parameter to imply some variations in the scaling of the latent code. It allows the network to have an increased efficiency for a wide bit rate range. We tune our network to run at around 2 bits per pixel with a range of around [1.4, 3] bits per pixel. In Figure 3.3, we compare several model train for specific rate-distortion points with a single variable model. We compute the BD-SNR and BR-RATE on the best single point model compared to the variable model. We obtain BD-SNR = -1.25, BR-RATE = 1.44%. The use of a variable model represents an additional bit rate cost of only 1%. These gain units allow a variable bit rate for a single model and act as a quality parameter that is added during inference while reducing memory consumption.



(a) Distortion measure: SNR

(b) Distortion measure: PSNR

Figure 3.3 – Our variable model compared to single rate-distortion models.

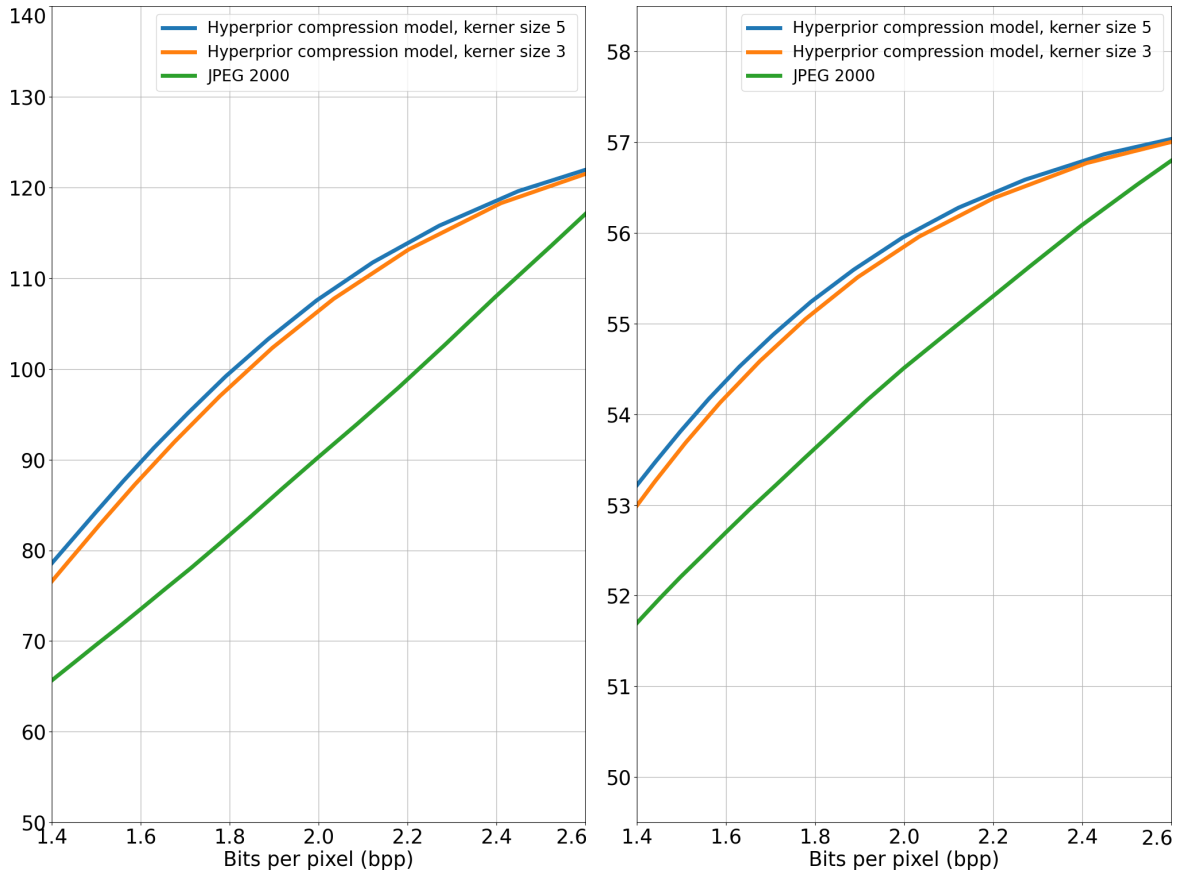
3.2.2 Combination of high bit rate and low complexity

The number of parameters to be learned during training is determined by the number of layers, their number of filters and the size of the convolution kernel. They all have an influence on the final size and complexity of the network.

3.2.2.1 Kernel size and network extraction capability

The original hyperprior architecture [Ballé et al., 2018] uses 5 x 5 kernel size for convolution in both the encoder and decoder parts. The hyperprior autoencoder has a filter size of 3 x 3. Work has been done to understand the effect of kernel size on compression performance [Alves de Oliveira et al., 2021]. The conclusion is that a 5 x 5 kernel support is sufficient, since a 7 x 7 kernel does not improve the approximation capabilities of the model, while 3 x 3 results in slightly lower performance.

However, looking at the performance for both the 3 and 5 square kernels in Figure 3.4, both rate-distortion curves are essentially the same with a relative difference between the rate-distortion curves of BD-SNR = -1.27, BD-RATE = 2.0%. This result may be due to the higher entropy contained in satellite images, which have more information per pixel than natural images, so the filter size does not need to be large to capture the information in the area of a pixel. This may explain why the 7 x 7 kernel did not give any improvement. In our case we are working with a 50 cm resolution image, so the entropy is even higher than in the 70 cm panchromatic data from [Alves de Oliveira et al., 2020, 2021]. A 3 x 3 kernel size may therefore be sufficient for a good approximation of the network with the advantage of being faster.



(a) Distortion measure: SNR (b) Distortion measure: PSNR

Figure 3.4 – RD curves for 5 x 5 and 3 x 3 hyperprior compression model.

3.2.2.2 Increase the network's capacity using the number of filters

The network consists of convolutional layers with N number of filters and the last layer, called the bottleneck layer is made of M filters. Models targeting high bit rates can suffer from saturation in their performance gains if their capacity (i.e. the complexity of the model in terms of the number of nodes and parameters) is not high enough for a given bit rate target, as shown in Figure 3.5. In this Figure from [Ballé et al., 2018], the authors do not discriminate between N and M. However, it has been shown in subsequent work [Johnston et al., 2019; Minnen & Singh, 2020] that a slight increase of the bottleneck layer while lowering the number of filters elsewhere lead to stable performance with an overall decrease of parameters.

Because we aim for a high bit rate, we increase the number of filters from 192 (encoder and decoder part) and 256 (bottleneck layer) to 256 and 448 respectively. We maintain this increased number of filters for all networks to mitigate the saturation effect at a higher bit rate. The bottleneck layer with 448 feature maps also allows the network to have the same order of magnitude in terms of total dimension compared to the input. Indeed, all data go through 4 convolutional layers of stride 2 so the number of parameters for each feature map at each layer is effectively divided by 4.

At the bottleneck, the dimension of each 448 feature maps is equal to the input dimension divided by 256. Given images of size n^2, we thus go from $3 * n^2$ pixels for the input data to $\frac{448}{256} * n^2$ dimension in the latent space. The model is at the limit of over-parametrisation as we keep roughly the same amount of information between the input of the encoder and the latent representation. With a much smaller bottleneck layer, we would lose data representation capabilities which is crucial in the context of high quality compression.

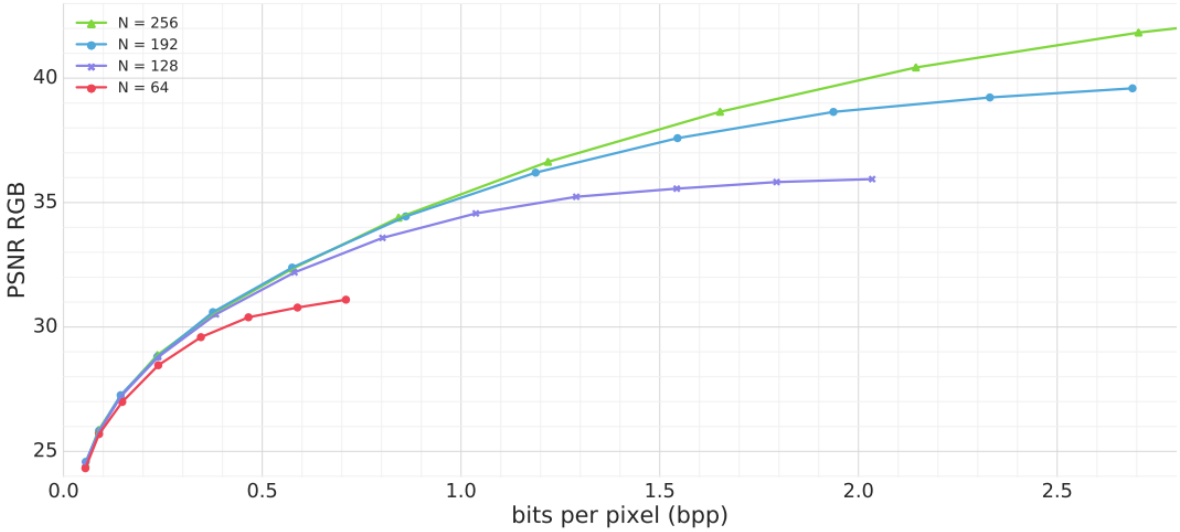


Figure 3.5 – RD curves for factorised-prior models differing in the number of filters N (No adjustment between N and the bottleneck layer M). Source: [Ballé et al., 2018].

3.2.2.3 Achieve GDN normalisation with fewer parameters

Generalised Divisive Normalisation layers (GDN) were introduced in Johannes Ballé's work [Ballé et al., 2016a, 2016b, 2017, 2018] as a normalisation transform in the field of image compression. In Equation (3.4) x_i and z_ii denote the input and output data. α_{ij}, β_i, γ_{ij} and ϵ_i represent trainable parameters. i and j are the channel index.

$$z_i = \frac{x_i}{(\beta_i + \sum_j \gamma_{ij} |x_j|^{\alpha_{ij}})^{\epsilon_i}} \tag{3.4}$$

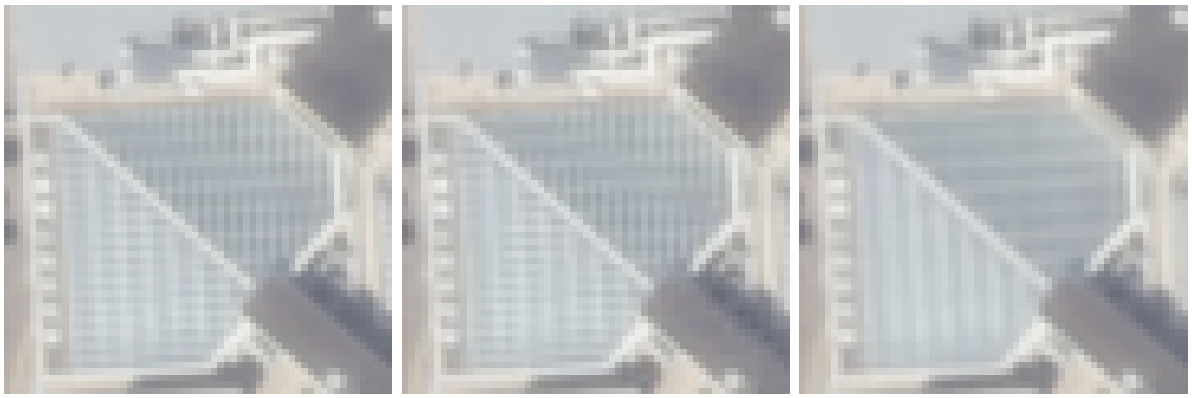
This normalisation layer proves to be very efficient in transforming the local joint statistics of the images into Gaussians. It shows great performance compared to traditional non-linearities at all bit rates [Johnston et al., 2019], but especially at high bit rates. The slight increase in the total number of parameters to be learned is outweighed by the improvement in rate-distortion. However, in our compression network, we will use a simplified version from [Johnston et al., 2019] that reduces the overall complexity with some fixed parameters ($\alpha_{ij} = 1; \epsilon_i = 1$) while maintaining equivalent rate-distortion performance. Those simplifications mostly remove complex square root computation. It suffers from a minor drop in performance for sensible gain in computational complexity. The GDN layer now becomes:

$$z_i = \frac{x_i}{\beta_i + \sum_j \gamma_{ij} |x_j|} \tag{3.5}$$

3.3 Towards better high-frequency reconstruction

We present several contributions that all aim at achieving better high-frequency reconstruction. The end-to-end compression model that we develop in Section 3.2 achieves higher rate-distortion performance than its traditional counterpart JPEG 2000. However, when we look at the visual quality of the reconstructed images, we see that some details have disappeared. In Figure 3.6, the compression model derived from the previous section is compared to JPEG 2000 at 2 bits per pixel with the ground truth. Overall, both compressed images obtained a high SNR as a consequence of a high bit rate, but our deep learning model, although performing much better on this pixel-based metric, has a poor visual quality on the high-frequency striped pattern of the image. All these high-frequency details with a spatial period of 2 pixels result in a high reconstruction error with neural network compression.

In this section, we first propose to augment data, with a special focus on diversity of striped pattern. To do so, we add more data augmentation during training with shear mapping transform to create more input data that are troublesome to compress. Second, we explore attention modules to highlight challenging parts in the image and balance the bit rate between these high-frequency details and texture. Any subsequent mention of *our model* will refer to the hyperprior compression network [Ballé et al., 2018] with all modifications from previous sections.



(a) Ground truth (b) JPEG 2000, SNR = 74 (c) Our model, SNR = 89

Figure 3.6 – Visual comparison of the compressed images (2 bpp) with the ground truth. Geometric resolution of 50cm. (c) is our model after the changes from Section 3.2.

3.3.1 Shear mapping of the training data

Data augmentation is already performed to increase the variability of the training dataset. This variability reinforces the representative and exhaustive nature of the training data set. In the long run, the increased data volume will result in better performing, more reliable models that are more resistant to variation. To induce rotational invariance in the data, we perform a random amount of $\frac{\pi}{2}$ rotation.

The problem is not the total number of images we trained our network on, as the overall compression produces good results. However, some small patterns are poorly compressed (e.g. striped patterns) and produce a high reconstruction error. Data augmentation through the use of a random amount of shear mapping (mapping based on a shear transform, also called transvection) paired with rotation is aimed at those challenging structures and increases the number of occurrences during training with a sub-pixel spa-

tial re-sampling. A shear mapping is a linear map that acts as a translation in perspective. It shifts each point in a fixed direction by an amount proportional to the distance from a parallel line passing through the origin.

The influence of shear mapping for better visual reconstruction is clear in Figure 3.7 with the striped patterns that are now visible. They tend to be blurred in deep learning models, although the overall compression achieves a higher SNR. This is illustrated in Figure 3.8, using the Fourier transform. Shear mapping allows the network to explore more of the spectrum and retain more high-frequency information, which is the case for 1-pixel striped patterns.

(a) Ground truth (b) No shearing, SNR = 89 (c) Shearing, SNR = 85

Figure 3.7 – Visual comparison of compressed images (2 bpp) with the ground truth. Geometric resolution of 50cm. (c) is our model with shear mapping.

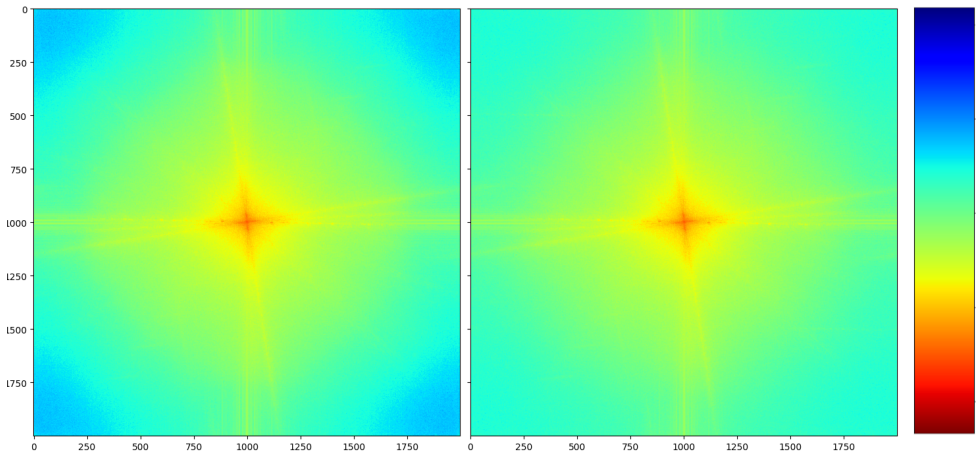
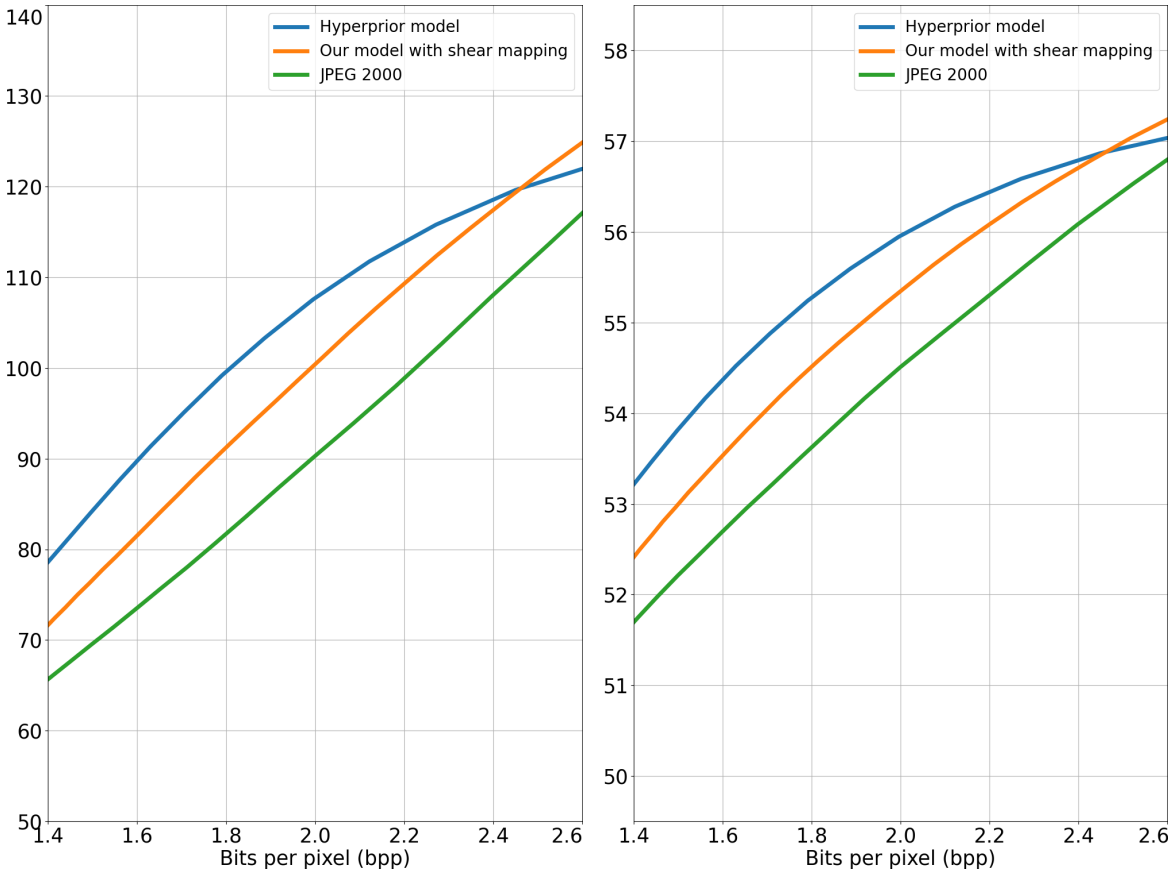


Figure 3.8 – Left: Our model, Right: Our model with shear mapping

When analysing the rate-distortion curves in Figure 3.9, we see that the hyperprior model has a good reconstruction overall but saturates at higher bit rates and fails to reconstruct high frequencies. Our shear mapping model provides a solution to this problem. The focus on these high frequency details is at the expense of a good reconstruction metric. Nevertheless, the visual quality provided by the shear mapping is significant, it helps to reduce overfitting and thus acts as a regulariser.



(a) Distortion measure: SNR (b) Distortion measure: PSNR

Figure 3.9 – Influence of shear mapping during training.

3.3.2 Attention modules to highlight challenging information

The use of attention modules [Woo et al., 2018] is motivated by the performance obtained in computer vision tasks. In the context of image classification, these layers are used to discard non-relevant background information. The learning of this trade-off for each dataset depends on the context and is driven by gradient descent. In the context of

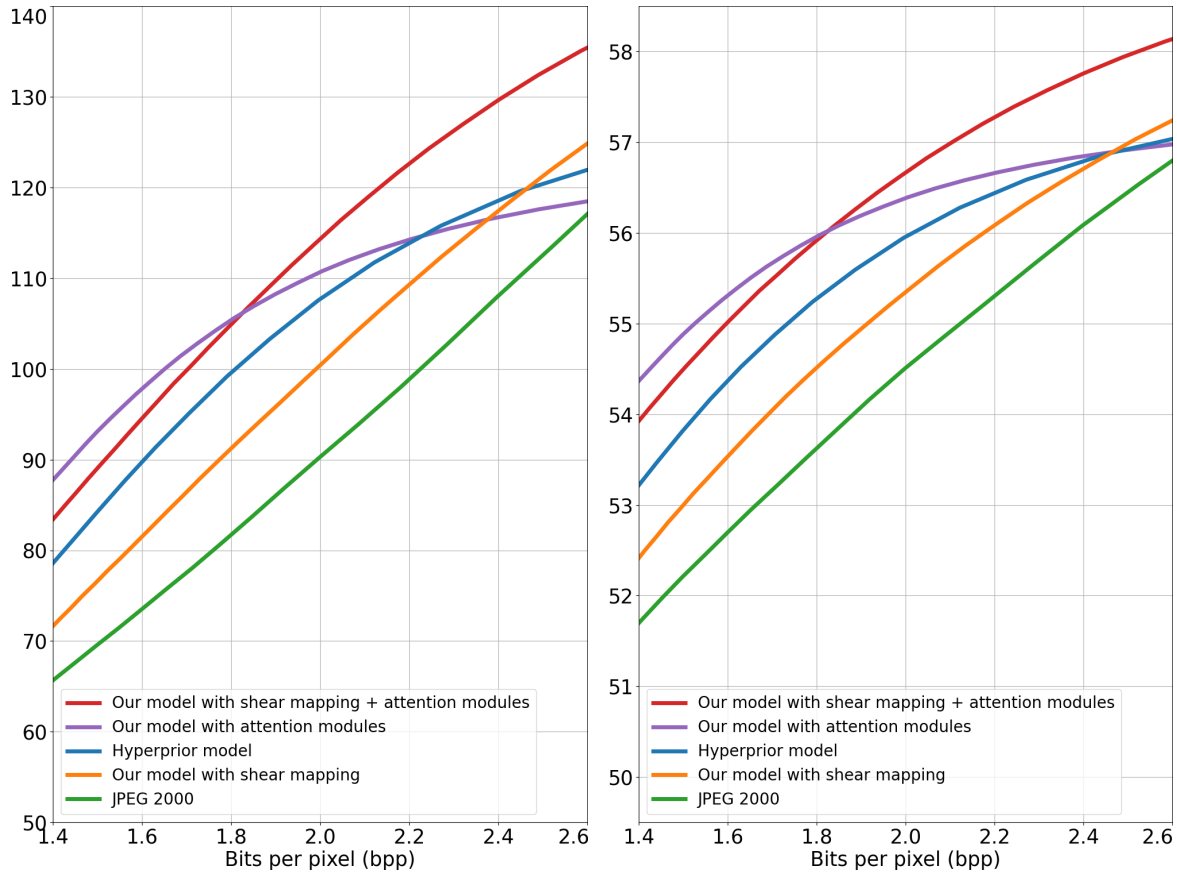
image compression, the attention mechanism guides the features and helps the network to highlight the challenging part of the image to balance the bit rate between edges, high frequencies and textures. We use a lightweight version [Z. Cheng et al., 2020] without non-local blocks that comes with a significant reconstruction gain for low computational complexity added.

(a) Ground truth (b) No shearing, SNR=101 (c) Full model, SNR=96

Figure 3.10 – Visual comparison of compressed images (2 bpp) with the ground truth. (b) is our model without shear mapping during training but with attention modules. (c) is our full model with both contributions.

We evaluate our model with shear mapping and attention modules combined, on a subset of representative satellite images in Figure 3.11. The attention-only model has an overall greater SNR at low bit rate but blur remains a compression artefact as seen in Figure 3.10. Also, it saturates at a high bit rate compared to other deep models. At low bit rate textures are well reconstructed but many high-frequency details are missing as the capacity is not large enough and the model saturate.

Our complete model which includes both attention modules and shear mapping, can mitigate the shear mapping downside and ensure a better reconstruction on average than the hyperprior model while preserving the high-frequency details. At high bit rates, our model does not saturate with the amount of details gained through data augmentation. A comparison with the hyperprior model gives the following results: BD-SNR = 7.75, BD-RATE = -8.5%. It results in an improved distortion performance compared to the current baseline without sacrificing much of its perceptual quality with challenging patterns.



(a) Distortion measure: SNR (b) Distortion measure: PSNR

Figure 3.11 – Effects of attention modules and shear mapping.

3.4 Improving the rate-distortion trade-off with a novel compression loss

The compression model designed in the previous sections shows better rate-distortion performance than the traditional baseline. The whole architecture is described in Figure 3.12 with all the changes from the previous sections. However, the aim was more to overcome the inherent limitations of deep learning model (lessened the blur added during compression) rather than to improve the hyperprior architecture. This section explores ideas to push our compression network towards better rate-distortion trade-offs by incorporating a perceptual loss function to extract structural information.

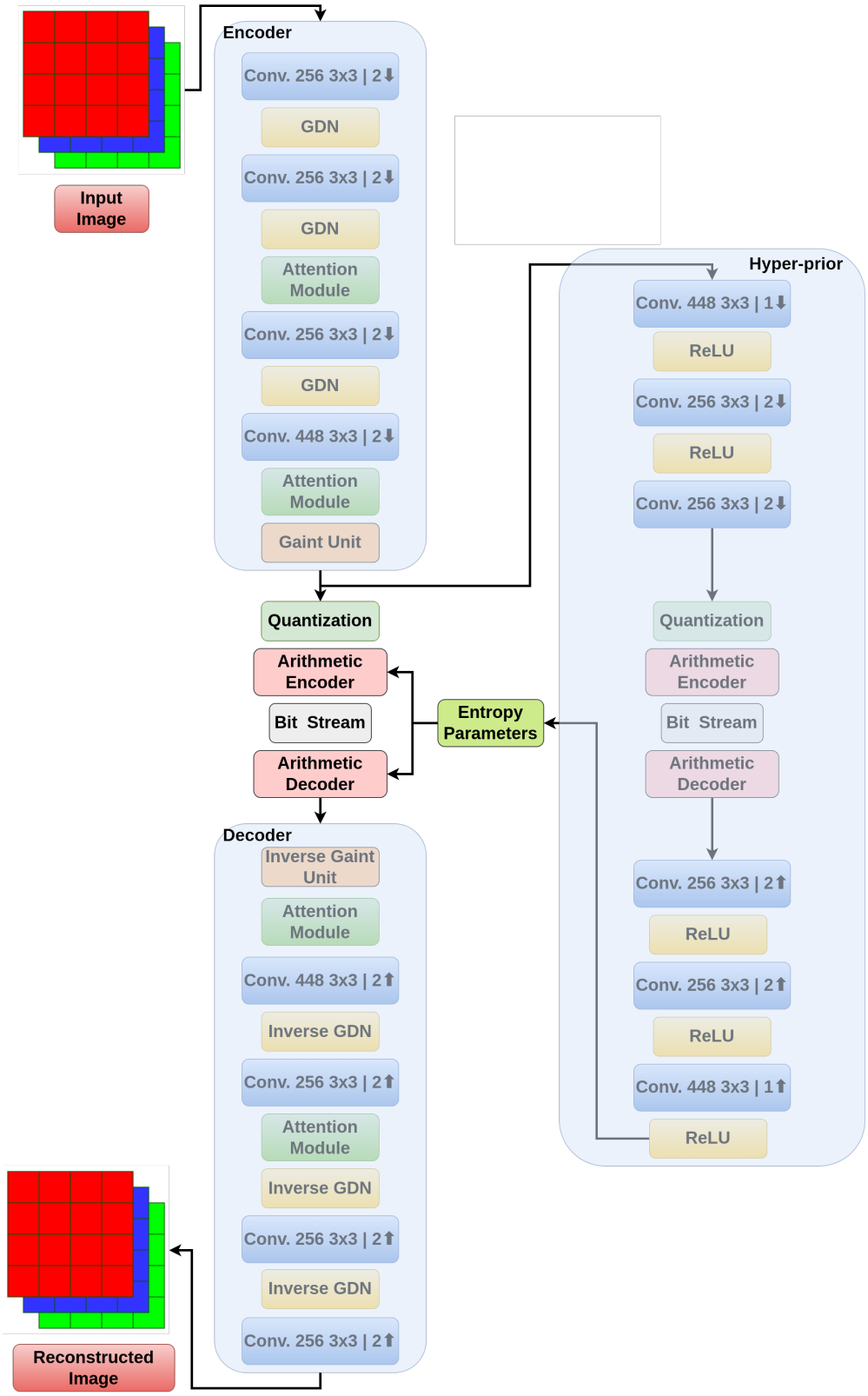


Figure 3.12 – Proposed architecture after changes to layers, filters and kernel.

3.4.1 Perceptual loss to extract low level spatial features

The main source of error in our reconstructed images comes from the blur generated during compression. It hides high-frequency striped patterns that have a pixel-sized spatial frequency. This behaviour is induced by the use of the $L2$ norm as the distortion metric during training [Y. Liu et al., 2021; Rad et al., 2019; Sajjadi et al., 2016; Yang et al., 2018]. To better adapt to the characteristics of the data and better reconstruct these highly typed areas during compression, we want to remove this unwanted effect.

To that end, we include an a priori that locates these areas of interest by adding a perceptual metric to the cost function to reduce the sole effect of the distortion metric. This metric differs from pixel-based metrics in that it aims to compute a distance between features extracted from the reconstructed $\hat{x}$ images and the x ground truth images. This perceptual loss is used in a wide range of applications [Johnson et al., 2016; Rad et al., 2019; X. Wang et al., 2018] to generate more realistic textures and sharper edges in image processing problems.

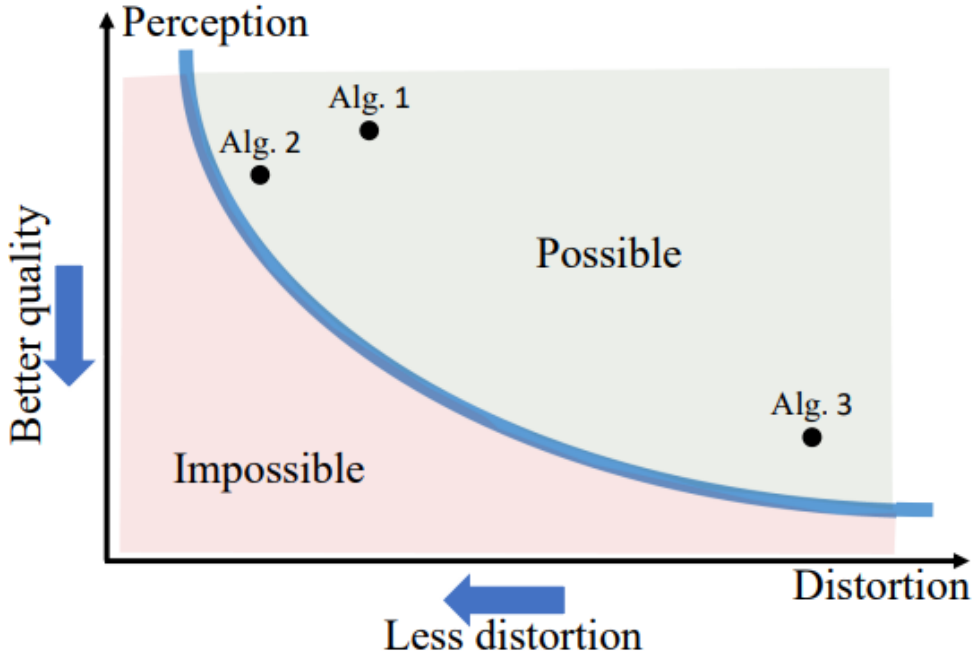
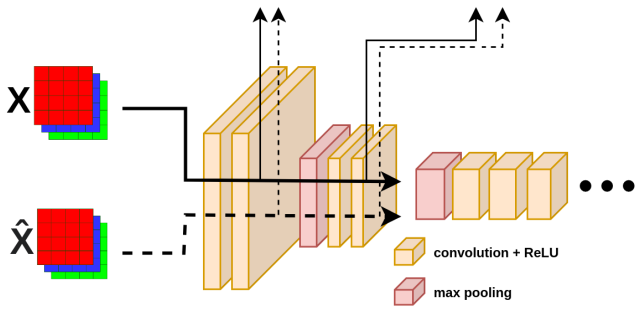


Figure 3.13 – The perception-distortion trade-off. Source: [Blau & Michaeli, 2018]

The choice of a perceptual metric is difficult as there is a wide range of options but we know that there is a trade-off between distortion and perceptual distance [Blau & Michaeli, 2018, 2019] as shown in Figure 3.13. This is analogous to the rate-distortion trade-off, which states that for a given compression algorithm there is a boundary that separates the infeasible from the achievable. For the perception-distortion trade-off, when at the boundary, improvement of one of the two characteristics is at the expense of the other.

From this work [Blau & Michaeli, 2018], we know that there is one metric for which this trade-off is less severe: VGG classification networks. We therefore define a cost function based on this network to extract structure within our features and guide learning towards better high-frequency reconstruction. We take a distance $L2$ on the latent representations of x and $\hat{x}$ at depths 2 and 4 of VGG. We define a loss function based on VGG [Simonyan & Zisserman, 2014] to extract structures within our features and guide learning towards a better high-frequency reconstruction.

$$P(x, \hat{x}) = \frac{1}{nm}(VGG_{0:2}(x, \hat{x})^2 + VGG_{0:4}(x, \hat{x})^2)$$



We choose to use only the early layers of VGG because they oversee the learning of low-level spatial features [Rad et al., 2019] while deeper layers focus on more abstract features. As our problem is more detail oriented, we use the first four layers of VGG to extract two sets of features. We compute the $L2$ norm between the ground truth and the reconstructed features.

We obtain a **rate-distortion-perception** trade-off:

$$\mathcal{L} = \lambda_a D(x, \hat{x}) + \lambda_b P(x, \hat{x}) + \alpha R(\hat{y}) \tag{3.6}$$

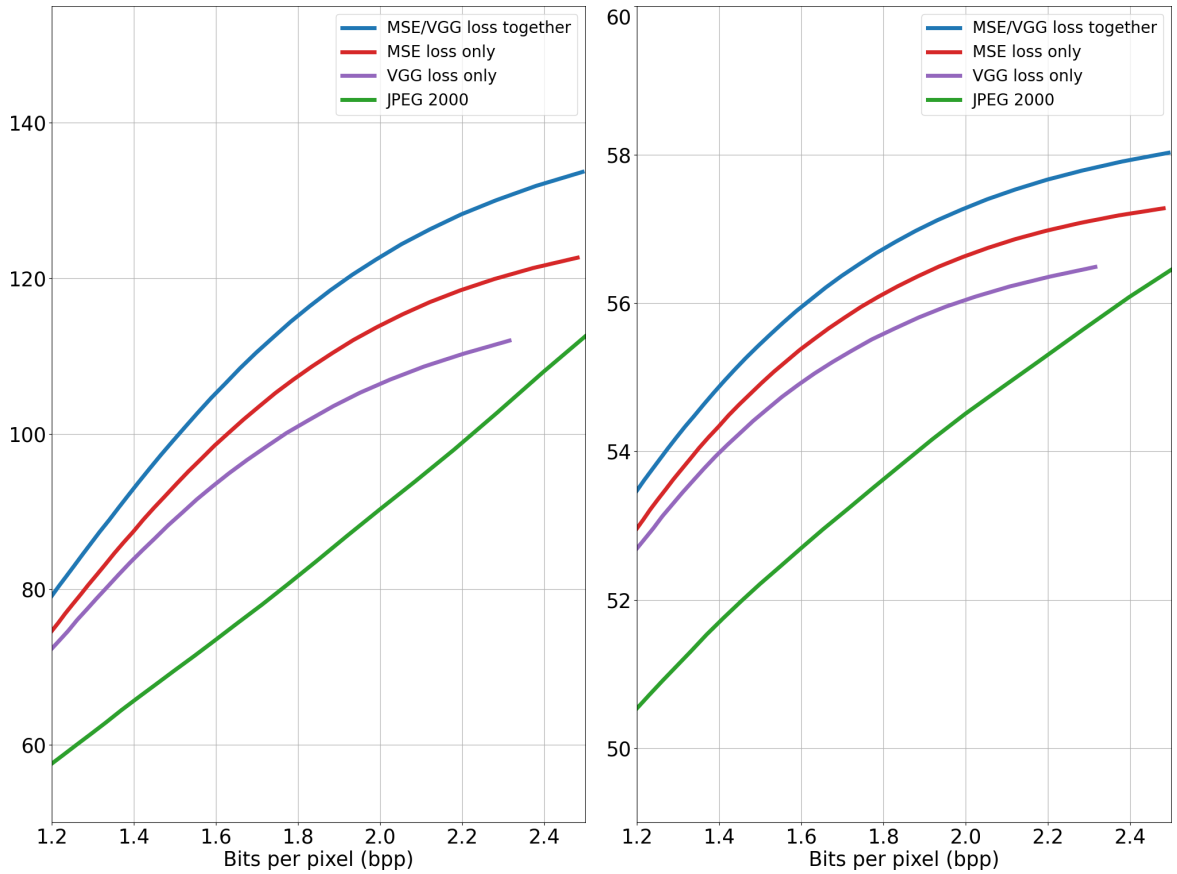
Both λ_a and λ_b are empirically set so that the distortion and perceptual metrics are of the same order of magnitude when the network has converged

$$\lambda_a = 2.6 * 10^6; \lambda_b = 10^4$$

α is set to target a bit rate and controls the rate-distortion trade-off.

We then assess the performance gain that perceptual loss brings to our compression model in Figure 3.14. We compare our model for different loss functions based on MSE, VGG or both losses when applied during training. VGG alone still performs well on the SNR and PSNR metrics even though it is not tailored to the MSE distortion, meaning that the information present in the VGG feature maps captures a perception of the pixel-wise distortion. When trained with MSE alone, the network performs unsurprisingly better when evaluated with using pixel-based metrics. Combining both metrics leads to even better performance, especially at high bit rates. By comparing the rate-distortion curve of both losses with MSE only we obtain: BD-SNR = 6.68, BD-RATE = -7.8%.

The idea that adding a loss that is not optimised for an MSE metric could lead to gains in both perceptual and distortion terms seems surprising. Indeed, according to the work of [Blau & Michaeli, 2018, 2019] and the Figure 3.13, if we were at the perception-distortion frontier then we would necessarily have had a compromise between these two cost functions. But we are probably not yet at the frontier of the best compromise, so we have been able to make progress on both fronts by adding a perceptual loss. We have improved the overall perception-distortion and rate-distortion performance, which on the diagram in Figure 3.13 is similar to going from Algorithm 1 to 2.



(a) Distortion measure: SNR (b) Distortion measure: PSNR

Figure 3.14 – Comparison of compression models with different loss functions.

3.4.2 Multi-loss balancing strategy

Learning for multiple tasks, with their respective loss functions, can lead to a better result for all tasks than learning for each task individually as shown in [Kendall et al., 2018; S. Liu et al., 2019] where semantic classification and depth estimation induce better performance together than separately. However, this implies adding more balancing terms in the loss function and since hyperparameters are troublesome to tune, it becomes harder to optimise the network. In our case we train for a single task (i.e. image compression) but with multiple loss functions (i.e. distortion and perception metric) which also requires hyperparameter tuning in the global loss function.

In our rate-distortion-perception trade-off formulated in equation (3.7), the rate is fixed during training to target a specific bit rate but both λ_1, λ_2 must be chosen arbitrarily.

$$\mathcal{L} = \lambda_1 L_1(x, \hat{x}) + \lambda_2 L_2(x, \hat{x}) + \alpha R(\hat{y}) \tag{3.7}$$

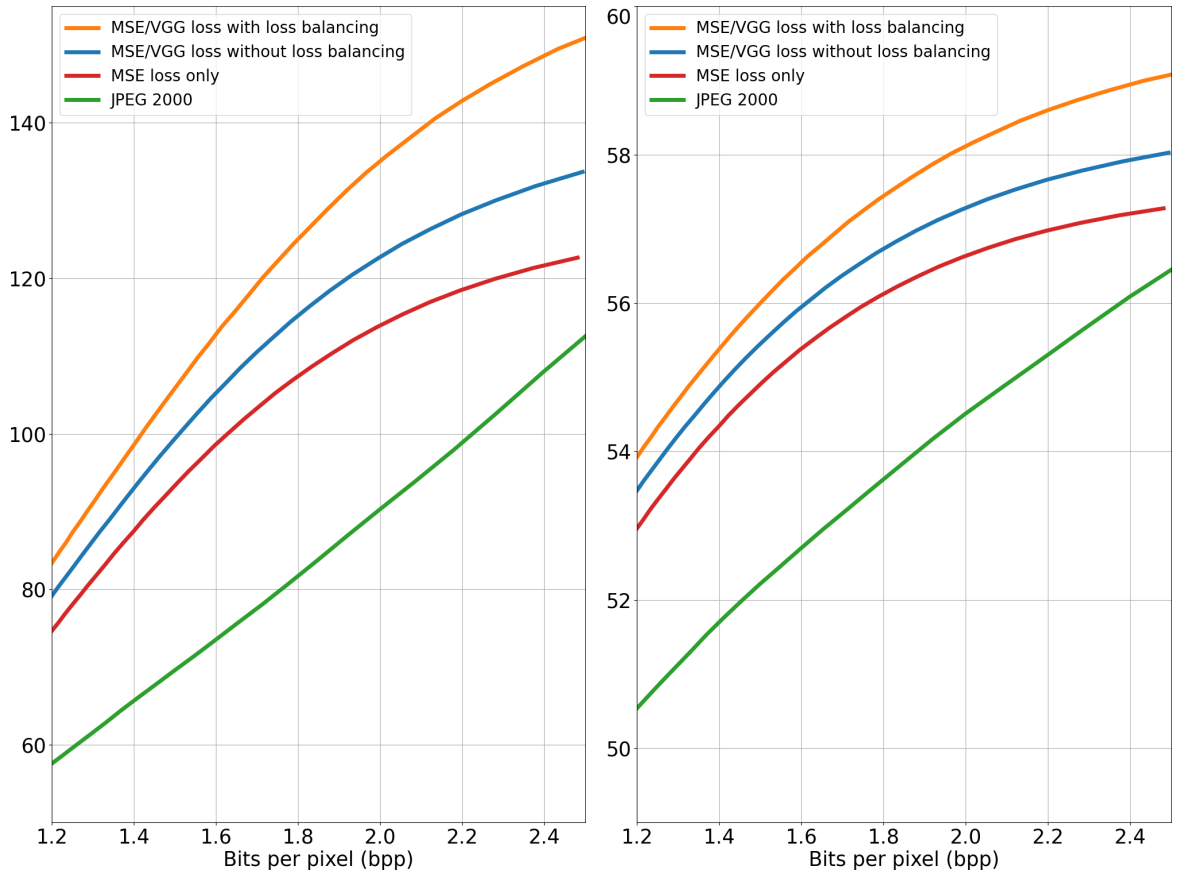
with $L_1 = \lambda_a D; L_2 = \lambda_b P$ set as in equation (3.6)

One solution to set the different loss parameters is to jointly tune all parameters within the loss function with an automatically controlled scheme [Crawshaw, 2020; Ruder, 2017]. It removes the need for manual tuning beforehand and ensures an optimal trade-off between all loss terms [Bischof & Kraus, 2021; S. Liu et al., 2019]. To automatically evaluate the λ_k we are following the dynamic weight average approach [Bischof & Kraus, 2021; S. Liu et al., 2019] to compute at each epoch a new λ_k based on previous loss measures for the distortion and perceptual metric: The loss function in equation (3.7) stays the same but now hyperparameters becomes:

$$\lambda_k(t) = 2.\frac{exp(\frac{w_k(t-1)}{T})}{\sum_i exp(\frac{w_i(t-1)}{T})}, w_k = \frac{L_k(t-1)}{L_k(t-2)} \tag{3.8}$$

Each loss L_k is associated with its corresponding λ_k. T measures the softness of the process by analogy with the annealing temperature and controls the proximity of different values of λ_k. For the first two epochs, we set w_k to 1. Then $\lambda_k(t)$ is computed using stored values of previous iterations of distortion and perceptual loss.

We compare the rate-distortion performance of compression models with the multi-loss balancing strategy. The multi-loss balancing scheme brings the previous model to a better rate-distortion trade-off over the whole bit rate range. Compared to the model without the multi-loss balancing strategy, we achieve BD-SNR = 8.8, BD-RATE = -8.4%. During training, the parameters λ_k adapt to the relative importance given to their respective task in previous epochs. If the value of one of the perception-distortion terms overwhelms the other then at the next epoch λ_k are changed according to their respective value so that the underrepresented loss term gains more relative importance. This enables the network to escape some local minima as the main λ_k leading the gradient changes over time.



(a) Distortion measure: SNR (b) Distortion measure: PSNR

Figure 3.15 – Comparison of compression models with different training loss.

3.5 Towards quasi-lossless compression

The progress in terms of rate-distortion performance achieved by our compression network is significant compared to the baseline hyperprior architecture and to JPEG 2000. Nonetheless, we are still in the scope of lossy image compression with our work and we would like to close the gap with lossless compression. In this section, we attempt to reduce patches of error that are still poorly reconstructed with a residual compression that targets exclusively high-frequency errors.

3.5.1 Dedicated compression network for high-frequency detail

Even with the improvement provided by the shear mapping and attention modules regarding high-frequency details and the rate-distortion gain achieved with perceptual loss, striped patterns are still amongst the most noticeable patches of error in our images. In Figure 3.16 textures and especially shadows cast by buildings have almost no error. Many areas have minimal error and no particular structure stands out. However, we can distinguish patches of a homogeneous error on rooftops and the striped patterns in pedestrian crossings were the error reconstruction is very high.

Figure 3.16 – Residual image after our neural network compression.

To solve this problem we will use two compression networks to compress a larger frequency spectrum. The reason is that instead of relying solely on a general compression network that inherently has troubles with high-frequency details, we prefer specialised networks, each one responsible for a determined part of the frequency spectrum. This leads to a generic heavy network for a general compression derived from previous sections and a secondary lighter network to capture high-frequency errors in the residual image as shown in Figure 3.17.

These errors resulted from a poor reconstruction of high frequencies in general (noise accounted), and striped patterns in particular. The network used to compress the residual is a lighter version of the general compression scheme with a reduced number of filters:

$N = 64, M = 128$. Nonetheless, as seen in Figure 3.16, most of the residual is unstructured noise. It means that very little correlation can be found in the image and it is very expensive to compress. We need to transform the image to only keep the structured noise that shows homogeneous patches of error. We filter the information contained in the residual image as pre-processing before using this in our specialised compression network.

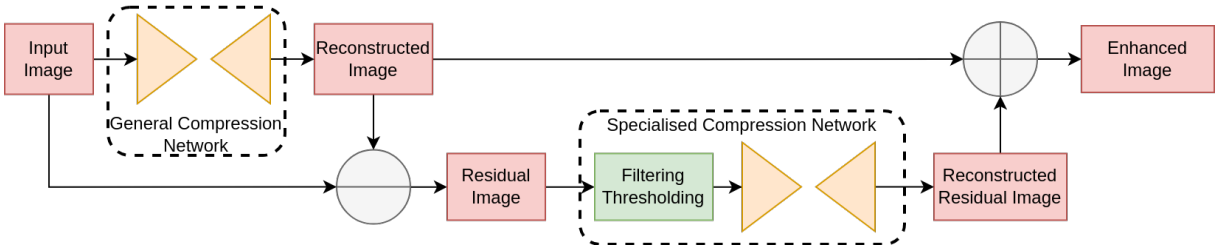


Figure 3.17 – Pipeline for the two compression networks

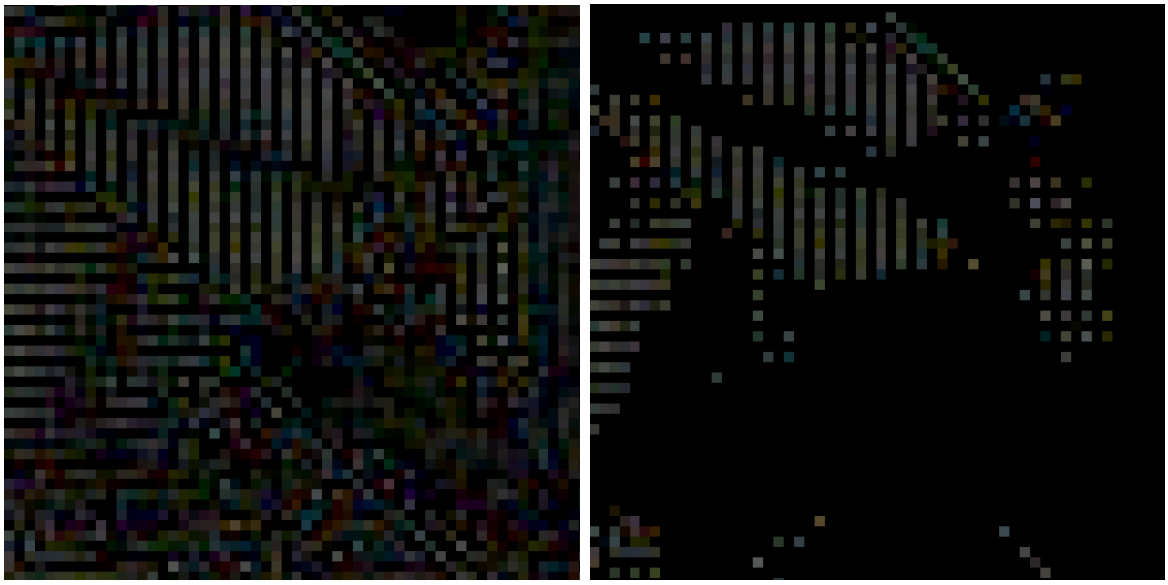
3.5.2 Filtered compression of the residual image

Not all of the residual image is compressed equally, as we use a mask to remove as much unstructured noise as possible. This mask is obtained by filtering and thresholding the luminance of the RGB residual image. High-frequency details are poorly reconstructed compared to textures, so we try to filter out random noise induced by compression and keep blocks of meaningful details. The following hand-crafted filter is used to retain only striped patterns.

$$Kernel = \frac{1}{14}\begin{pmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 2 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

The images we are working on, have a 50 cm spatial resolution and it results in having details at a pixel level. So we focus on patterns that have a 2-pixel period as from experience they are the ones that suffer the most from poor reconstruction. This is displayed in the filter with a spacing of each component. This filter highlights areas of a certain pattern at the expense of the other. We threshold the result to retain homogeneous patches of errors. Finally, this mask is applied to the original residual image to remove unstructured noise which is expensive to compress.

In Figure 3.18, we observe the effect of our processing on the residual images. We distinguish easily between random and structured errors. The whole process of filtering and thresholding is to highlight those meaningful areas that have not been properly compressed so that the specialised compression network only focuses on those few error patches.



(a) True residual before processing (b) Post-processing residual

Figure 3.18 – Effect of our filtering/thresholding mask on residual error

We analyse the effect of the combined general and specialised network on the visual quality in Figure 3.19. Even with the improvement of the high-frequency reconstruction from Section 3.3 in Figure (c), the pedestrian crossing still suffers from the blur caused by the compression and the details of the alternating stripes. In blue circles, we can see from the error map of the upper part of the images that textures that had artefacts with the JPEG 2000 codec are well recovered by the learned models. In red circles, we can see that high-frequency details are now reconstructed in the image (d) with the addition of the residual compression on top of the general compression.

Quantitative results of the effect of the residual compression are not shown because its impact on the overall compression performance is limited compared to the general compression network. In quantitative terms, the specialised network will increase the performances up to 2 SNR for a given bit rate for an increase of 0.3 bits per pixel. As seen with the various visual comparisons, the gain of this secondary compression network is more about locally recovering high-frequency details than globally enhancing the rate-distortion performance. Reconstruction is further improved by the addition of a second network dedicated to compressing the residual image. This compressed residual enables the network to better preserve high-frequency detail while effectively compressing textures.

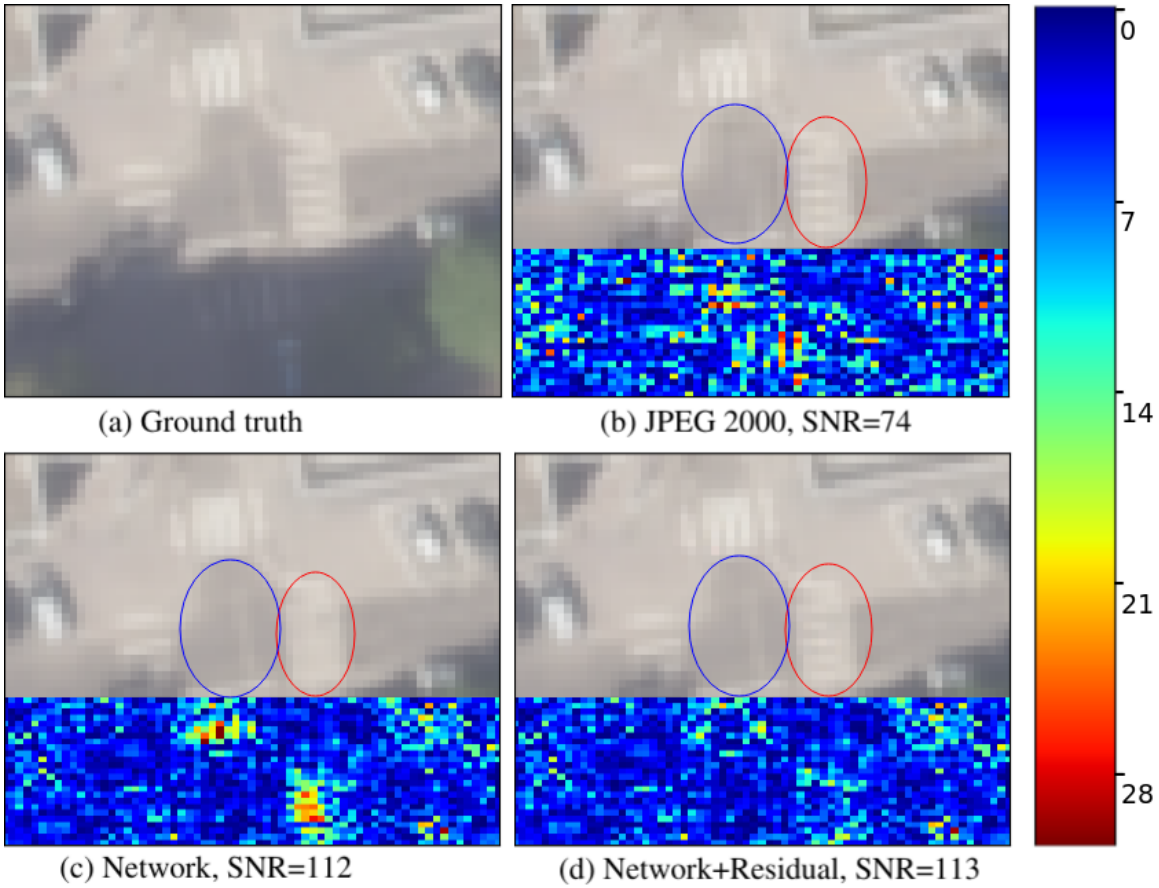


Figure 3.19 – Visual comparison of compressed images (2 bpp) with the ground truth. Error map of the upper areas of the compressed images with range (range value [0,32])

3.6 Conclusion

This chapter proposes several solutions to the problems of satellite image compression. Through a series of improvements, we address the issues of rate-distortion performance, high-frequency reconstruction, and adaptability to the use case and hardware environment.

We first adapt a reference learned image compression neural network [Ballé et al., 2018] to the specific use of RGB satellite image compression. We propose an architecture that take into account the higher bit rate needed for very high reconstruction quality. We analyse the number of filters and the size of kernels to obtain a trade-off between network complexity and rate-distortion performance. We enhance the network with gain units before quantisation to allow for an adaptable network that effectively works on a large bit range. We add supplementary layers and data augmentation to mitigate the saturation behaviour of the network which cuts out high-frequency details. This takes the form of attention modules and shear mapping during training to better highlight challenging parts of the data (i.e. striped pattern).

We then focus on improving the rate-distortion performance of our compression network. To achieve this, we move from the classical rate-distortion trade-off to a rate-distortion-perceptual trade-off by including a perceptual loss. The results are further refined by optimising the hyperparameters using a multi-objective strategy. Finally, we explore an extension to quasi-lossless compression, which takes the form of a multi-stage network. A general compression network is combined with a specialised network focused on compressing the residual image. This residual is processed to compress only meaningful error patches and not the compression noise. It achieves a marginal error of reconstruction for every piece of information in the data.

The proposed deep learning framework has an improved distortion performance that outperforms the CCSDS lossy compression standard [CCSDS, 2017] or JPEG 2000 without sacrificing its perceptual quality with high-frequency details. It addresses the challenges of satellite imagery with improvements in performance, operating conditions and versatility.

Chapter 4

State of the art for joint compression, denoising and demosaicking

Contents

4.1 Introduction

4.1.1 Background on the digital imaging pipeline

The standard image processing pipeline used in most digital imaging cameras consists of 3 main steps, shown in Figure 4.1: demosaicking, denoising and compression. The output from a camera sensor needs to be processed in order to achieve a correct rendering and for subsequent applications.

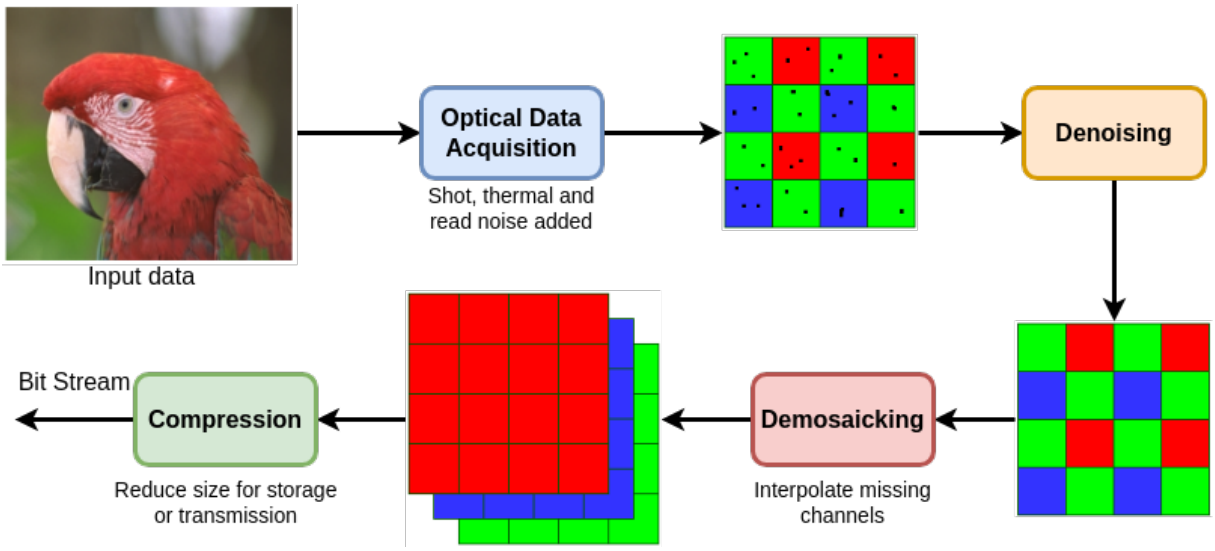


Figure 4.1 – Simplified imaging processing pipeline.

The majority of digital sensors today are CMOS image sensors due to their attractive cost and performance. These sensors are photodiodes that convert the light they receive into voltage to measure the intensity of the light. However, they only capture the monochrome intensity of the light, not the colour spectrum. To overcome this problem, colour filter arrays (CFAs) are stacked on the photodiodes of the sensors. There are many CFA profiles, but the most popular is the Bayer CFA, which consists of two green pixels diagonally, one blue and one red, as shown in Figure 4.1. The spectral sensitivity of the human eye is highest at the red, blue and green wavelengths, with a peak at the green wavelength, which explains the emphasis on two green pixels for each of the other colours. Each pixel of the raw image is either red, green or blue depending on its Bayer profile. The demosaicking process is used to recover the missing information of each colour channel by interpolation. Standard demosaicking methods include bi-linear interpolation [Losson

et al., 2010], Malvar algorithm [Malvar et al., 2004] and Hamilton-Adams demosaicking [Buades et al., 2011b].

However, the signal that needs demosaicking is not noiseless as several phenomena introduce noise in the early processing pipeline. Sensor acquisition adds some shot noise due to the discrete nature of the photons (Poisson noise, significant at high light intensity). Photodiodes are also affected by the thermal noise of the electric charges (Gaussian noise, independent of the intensity) and finally, some read noise can be caused by the electronics inside the camera (important for low-intensity signal). Standard denoising algorithms are BM3D [Lebrun, 2012] or non-local Bayes [Lebrun et al., 2013].

Denoising is necessary to support other processing, particularly compression which is used so that the signal can be stored or transmitted using less storage or bandwidth. Images are signals with many spatial and spectral correlations that codecs use to reduce the number of bits needed to encode the signal. Since noise adds entropy to the signal, it makes it harder to compress, while degrading the information contained in the signal. Denoising is therefore as important for recovering a quality signal as for compressing it. The most commonly used compression algorithms are JPEG [Wallace, 1992] and JPEG 2000 [Skodras et al., 2001] using respectively a discrete cosine and wavelet transform.

Other processing includes white balancing, gamma correction, colour transformation and gamut mapping [Ramanath et al., 2005], but we want to focus on the three main steps shown in the Figure 4.1: demosaicking, denoising and compression.

4.1.2 Motivation

Most of the literature treats these three highlighted problems independently. Each problem has been the subject of extensive research for a long time, and the advent of deep learning has transformed the results achieved so far. For the compression task, Chapter 3 gives an overview of deep learning methods that have caught up with the best codecs [Bross et al., 2021; Sullivan et al., 2012] in just a few years. The pioneering work of [K. Zhang et al., 2017] with the DnCNN network is now the backbone of many state-of-the-art denoising networks [Tian et al., 2020a, 2020b; K. Zhang et al., 2018]. For demosaicking, convolutional networks have rapidly replaced hand-crafted methods to solve this ill-posed

problem with improved reconstruction performance [Syu et al., 2018; R. Tan et al., 2017; D. S. Tan et al., 2018].

However, there are several advantages to tackling these problems together rather than sequentially. First, the hypothesis of many methods is not supported by reality. Demosaicking is often performed assuming a noiseless image, but denoising does not necessarily occur before demosaicking, and residual noise remains after processing. Similarly, compression almost always considers images that have already been demosaicked and denoised, whereas the vast majority of the raw data are colour filter arrays. Combining these processes can also be beneficial for each of them. No further degradation is added to the signal if all processes are performed together and the computational resources are shared on a single task.

4.2 Joint denoising and demosaicking

The best order to perform denoising and demosaicking sequentially is still an open question, explored in recent studies [Jin et al., 2020]. The advantages of each method are as follows:

— Denoising first:

1. The noise is not yet correlated before demosaicking and is closer to the independent identically distributed assumption.
2. This assumption still holds for non-Gaussian noise present in images, such as Poisson noise with an Anscombe transform [Anscombe, 1948].
3. The best demosaicking algorithms are designed for noiseless images.

— Demosaicking first:

1. Processing on full-resolution images with fewer details lost compared to 4 subsampled channels.

A joint solution can solve this problem and provide several benefits. One of them is to include the presence of noise inside the demosaicking. It also reduces the overall computational complexity compared to sequential processing.

Methods can be divided into two types: optimisation-based with hand-crafted priors, and learning-based methods. We will give a brief overview of the methods used before the emergence of deep learning for joint denoising and demosaicking problems. We will then focus on neural network methods and conclude with networks that simulate the ISP pipeline with the addition of other typical image processing.

4.2.1 Non network-based solutions

Hirakawa proposed a first joint demosaicking and denoising method based on a hand-crafted filter of the Bayer filtered image [Hirakawa & Parks, 2006]. He added additional constraints to the CFA and used total least square denoising while recovering the full colour image. The results showed efficient joint processing with improved reconstruction.

Condat introduced a demosaicking algorithm for noisy images using a method designed for frequency channels. It first decomposes the image into frequency channels and then performs demosaicking on the luminance only [Condat, 2010]. This has been further improved with a total variation minimisation problem that includes an additional constraint on the raw image [Condat & Mosaddegh, 2012]. This enforces a closer reconstruction of the chrominance.

Some authors, such as Chatterjee, have focused on low-light images for their joint demosaicking and denoising framework. The main idea is to up-sample the noisy image and perform denoising on this representation, which is then enforced to the desired full colour image [Chatterjee et al., 2011]. Based on these previous works, Tan uses additional priors on the CFA image through the ADMM method (alternating direction method of multipliers) [H. Tan et al., 2017]. All of these methods achieved higher reconstruction performance than their respective sequential baselines at the time.

4.2.2 Deep learning framework

The use of machine learning in this joint processing approach began with the work of [Khashabi et al., 2014], which consists of random tree fields - a kind of decision tree. But it was really the work of [Klatzer et al., 2016] and [Gharbi et al., 2016] that paved the way for these joint neural network models. The former uses a sequence of demosaicking blocks similar to residual blocks with a connection between the output and the input. Each block

is responsible for an energy minimisation problem whose parameters are learned with a standard loss function over the $L2$ norm distance between the RGB ground truth and the reconstructed image.

4.2.2.1 First deep joint network

The work of [Gharbi et al., 2016] first introduced the use of deep learning with convolutional networks for joint processing. Their contribution lies in two aspects: architecture exploration and dataset generation. The input data is a noiseless raw image, which is then sub-sampled into the 4 channels of its CFA pattern with an additional noise channel to help the network recover noisy information, as shown in Figure 4.2. This noise channel can either be for a fixed σ to specialise the network on a particular noise, or it can vary in a continuous range of σ to provide better average denoising performance. These 5 channels are then fed into d convolutional layers and a final residual layer. These feature sets are up-sampled into 3 channels and form a 6-channel tensor with the addition of the input, which is separated into its corresponding 3-channel representation. A final convolution occurs before a final output into the 3 channels RGB enforced by an affine combination of the feature maps.

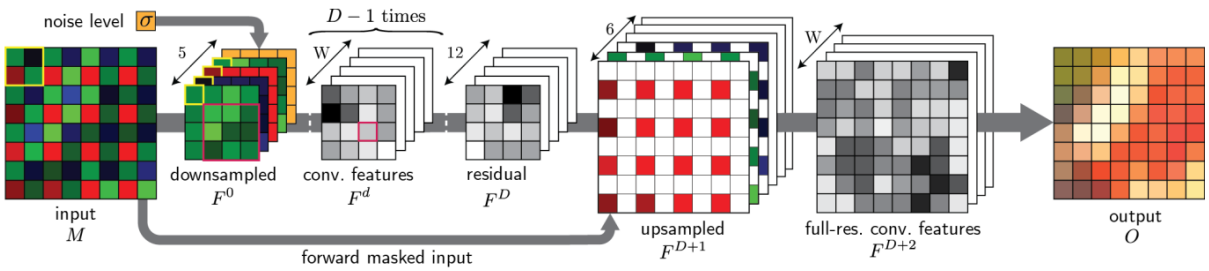


Figure 4.2 – Joint denoising and demosaicking architecture. Source: [Gharbi et al., 2016]

Because machine learning algorithms are data-driven, they rely heavily on the input data to extract information. The authors first trained their network on the millions of images contained in Imagenet [Deng et al., 2009], only to find that patches had a large variability in reconstruction quality. To be more precise, about 30% of the patches were difficult to process with a PSNR below 30, while the rest of the patches could be reconstructed easily with a PSNR above 40. They create their own subset of all the training images to include a larger proportion of difficult images in the training dataset. They

detect them by using a network trained on the Imagenet dataset and then analysing the poorly restored patches. The most prevalent artefacts in their reconstruction were salient luminance artefacts and moiré artefacts. Their smaller dataset still provides a better PSNR compared to other demosaicking methods, but this is all the more significant on datasets that target these difficult patches.

4.2.2.2 Iterative networks for joint demosaicking and denoising

Following in the footsteps of iterative optimisation strategies such as [Klatzer et al., 2016] briefly discussed in a previous section, Kokkinos proposed an iterative network that solves the inverse problem with a succession of residual blocks to answer the joint problem [Kokkinos & Lefkimmiatis, 2018, 2019]. The raw image is first convoluted to be a 3-channel input (or interpolated using bi-linear interpolation) and is fed to the network described in Figure 4.3.

Residual blocks [K. He et al., 2016] follow one another with the same input and output dimension so that the last output is already a 3-channel image. They allow better gradient backpropagation, as the skip connection reduces the effect of vanishing gradients in the deep architecture. In this noise-adaptive method, these layers are accompanied by the addition of noise as side information to help the residual layer extract and then remove the noise from the input. This behaviour is repeated many times throughout the network, meaning that the network can be tuned for a certain amount of noise removal depending on the processing resources.

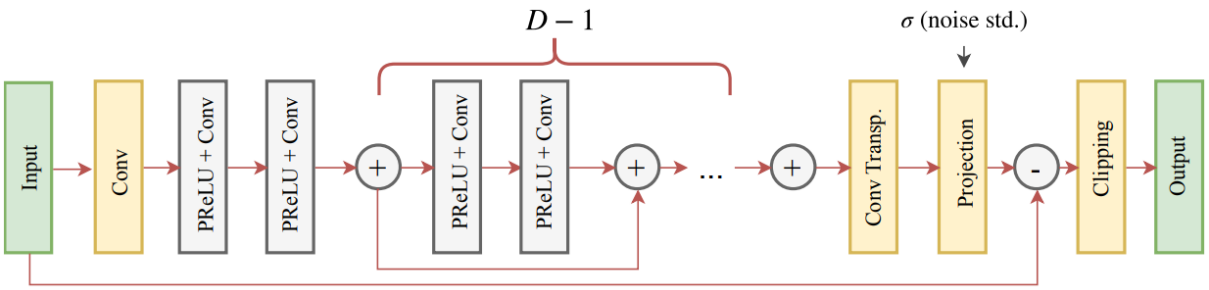


Figure 4.3 – Joint denoising and demosaicking architecture with a succession of residual blocks. Source [Kokkinos & Lefkimmiatis, 2019]

4.2.2.3 Self guidance for demosaicking

The reconstruction performance of the previous joint demosaicking and denoising networks was already better than sequential processing, but the work of [L. Liu et al., 2020] has further improved the result of such architectures. The main contributions of the authors are the novel loss functions for parameter learning and the architecture that takes into account the peculiarities of the CFA pattern of the raw image.

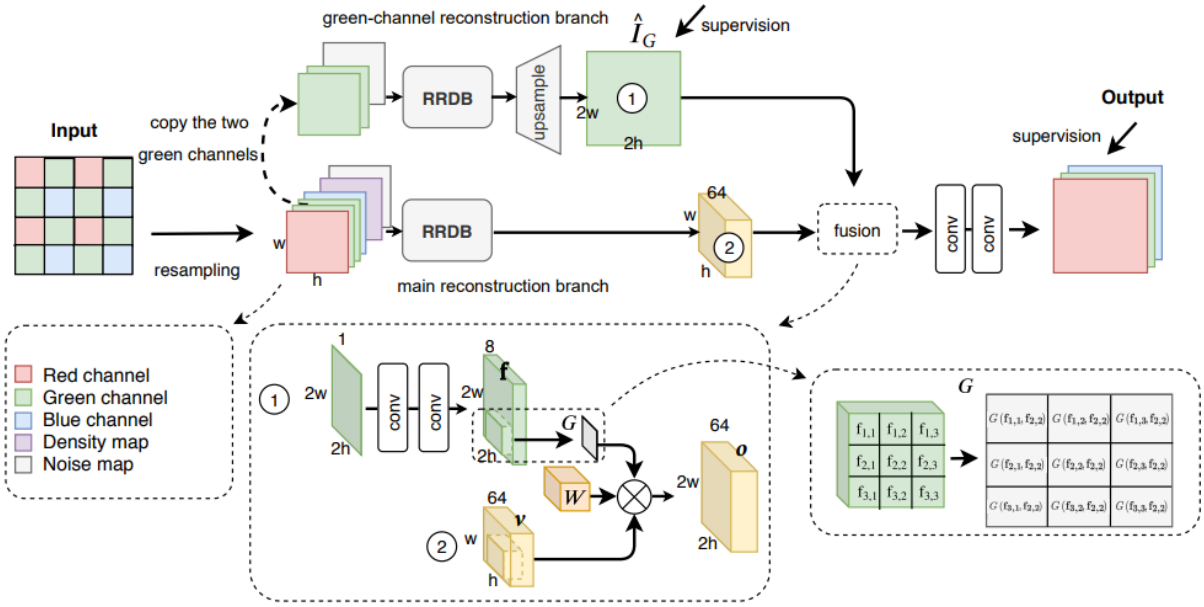


Figure 4.4 – Self-guidance architecture. Source: [L. Liu et al., 2020].

The authors sub-sampled the raw image, similar to other demosaicking methods, with the additional noise channel, but also with a density map M_D. The aim is to make the high frequencies stand out more clearly in a dedicated channel during training. In fact, neural networks tend to blur these high-frequency details, as they are often trained using only the $L2$ or $L1$ norm. The density map has a high value if the corresponding raw image has high-frequency details.

$$M_D = h * (g_2 * (I_G - g_1 * (I_G)))$$

Where h is a normalisation function, $g_1; g_2$ is a Gaussian blur kernel, and I_G is the grey image formed from the average of the four sub-sampled RGGB channels of the raw image.

The second contribution is to use the two green channels within the network and reconstruct them beforehand to help reconstruct the other channels. In the Bayer CFA, the red and blue channels are sub-sampled twice as much as the green channel to account for the eye's greater sensitivity to this wavelength. They use similar layers as in super-resolution problems to recover a full-resolution denoised green channel, which is fused with the main reconstruction branch using a spatially adaptive convolution operation. The result is resized to the output of one colour. The final contribution that differs from the previous demosaicking network is the implementation of additional loss functions to minimise the impact of blur in the reconstruction. Their purpose is to increase the attention of the network to high frequencies.

$$\mathcal{L} = L_{edge} + \lambda_1 \mathcal{L}_{smooth} + \lambda_2 \mathcal{L}_{L1} + \lambda_3 \mathcal{L}_{Green}$$

1. $\mathcal{L}_{edge}$ an adaptive-threshold edge loss
2. $\mathcal{L}_{smooth}$ an edge-aware smoothness loss
3. $\mathcal{L}_{L1}$ the $L1$ loss between the output and ground truth image
4. $\mathcal{L}_{Green}$ the $L1$ loss between the ground truth of the green channel and the green channel in the guidance branch

The adaptive-threshold loss is based on the Canny edge filter. It is applied to both the output and the ground truth which are then separated into patches. With the edge detector, the authors make an estimate of the number of pixels categorised as edges and deduce a loss function that can be assimilated to an edge amount detector. Areas of the image that contain many edges will induce a larger cost than textures. The edge-aware smoothness loss has a slightly different purpose since it tries to preserve the information edges while also smoothing noise in texture areas. It is a total variation loss with an exponential smoothing term. The resulting network achieves much higher PSNR for any noise level and many different datasets.

4.2.3 Emulate ISP pipeline

ISP stands for Image Signal Processing or Image Signal Processor. It is the chain of processes (or the processor responsible for them) used in modern digital cameras, as shown in the Figure 4.5. The full ISP pipeline includes processing such as white balance, gamma correction, colour transformation and gamut mapping.

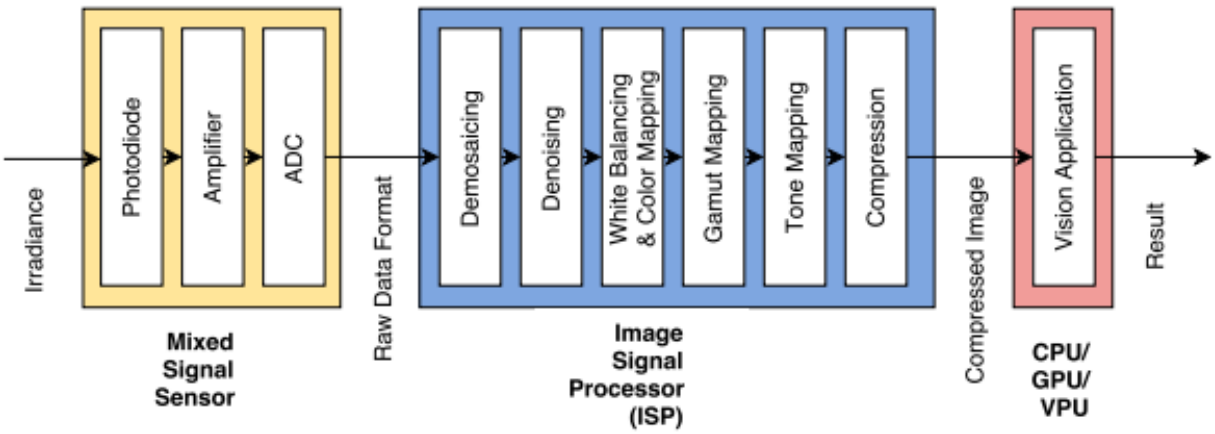


Figure 4.5 – Standard image processing pipeline. Source: [Buckler et al., 2017].

A considerable amount of work has been done to include a number of additional treatments that can be found in the pipeline with joint demosaicking and denoising. The work of [Ratnasingam, 2019] includes in their joint demosaicking and denoising network white balancing, exposure correction and defect pixel correction with additional prior in the form of a loss function to aid the reconstruction process. They also showed the effectiveness of their method with other CFA than the classical Bayer filter.

Working on different types of modality is also something that [Schwartz et al., 2019] and [A Sharif et al., 2021] are exploring. The former uses only images from a Samsung S7 camera to perform image restoration (as well as demosaicking and denoising) on low-light images. The advantage of these data-driven methods is that they extract the information contained in the image, such as the acquisition noise, which depends directly on the sensor. The latter adapts joint processing to the pixel-bin sensors found in smartphone cameras to overcome material limitations.

Additional processing such as super-resolution is performed by [Qian et al., 2019]. This processing is often beyond the scope of ISPs embedded in cameras but is combined in their work with demosaicking and denoising to help low-resolution sensors achieve higher performance. For the sake of generality, all of those processing are beyond the scope of our work, and we focus solely on the joint processes of demosaicking, denoising and compression

4.3 Joint compression and denoising

Images from sensors suffer from a certain amount of noise. This degradation can be amplified by any processing step in the imaging system pipeline that does not take it into account. This is particularly damaging for the compression step used to process noisy images, as noise is often a high-frequency type of information that is expensive to compress. This unwanted information in the image competes with other high-frequency details, reducing the performance of the rate-distortion trade-off.

Early literature on this topic revolves around thresholding coefficients in the wavelet domain [Chang et al., 2000] for codecs such as JPEG 2000. In the early 2000s, methods were developed to improve this type of process with Markov random field to compute the map of the oriented wavelet transform [Chappelier & Guillemot, 2006]. Joint optimisation of the distortion brought by compression and denoising in imaging pipelines has been explored for increased rate-distortion performance [Carlavan et al., 2012]. We have seen the rise of deep neural networks as part of the state-of-the-art in many image processing tasks including compression with the hyperprior architecture [Ballé et al., 2018] or denoising with the *DnCNN* [K. Zhang et al., 2017] network.

The work of Gonzalez [Gonzalez et al., 2018] proposed a joint compression and decompression in the wavelet domain but uses a CNN as a regularisation method to better extract the image statistics instead of patch-based methods. This evolution of neural networks now includes both processes in a joint end-to-end manner to merge denoising with compression, as in the work of [Brummer & De Vleeschouwer, 2023], which shows better rate-distortion results when the training is aware of the noise statistic. The core of the process is based on the extraction of noise statistics by the compression network during training on a pair of ground truth and noisy images.

4.3.1 Satellite image compression and denoising

Most of the state-of-the-art in deep learning joint compression and denoising is based on autoencoder networks and more precisely on the hyperprior architecture developed in [Ballé et al., 2018] and detailed in Chapter 2. This is the case of the work in [Alves de Oliveira et al., 2022], which specialises in panchromatic satellite imagery. The type of noise in these images is a mixture of Poisson and Gaussian noise and is highly sensor

dependent. A data-driven method such as neural networks removes the need for an a priori on the data. In the case of satellite imagery with a lot of shot noise, the data does not need to be processed with an Anscombe transform to make all the noise in the image Gaussian, so that standard denoising methods such as BM3D or non-local Bayes can be used.

Their architecture is presented in Figure 4.6 and their test includes a modified version where the hyperprior is replaced by a Laplacian entropy model developed in [Alves de Oliveira et al., 2021] which was shown to work with a marginal loss in performance for the compression-only problem, since features in the latent representation for satellite images are mostly Laplacian distributed. However, with noise degradation, this simplified entropy parameter estimator no longer gives good results.

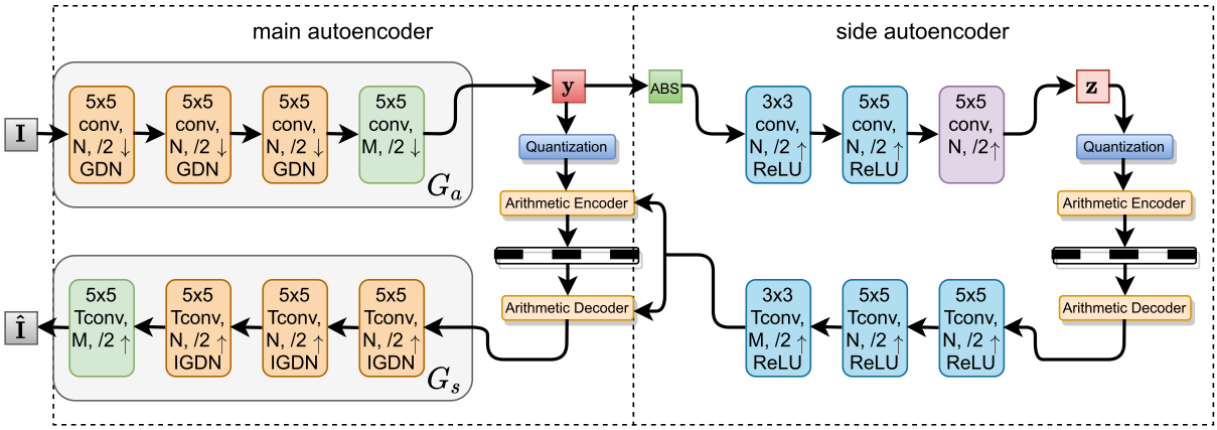


Figure 4.6 – Joint compression and denoising architecture. Source: [Alves de Oliveira et al., 2022].

They train their joint model on pairs of ground truth and noisy data and compare its rate-distortion performance with the sequential model using BRDNet [Tian et al., 2020b] as denoiser and the joint model with an additional denoising step on decompressed images with the same deep learning denoiser. The joint model achieves a reduced complexity compared to any sequential model using autoencoder based compression while maintaining on par performance with state-of-the-art sequential compression and denoising method. Further denoising can be done afterwards to enhance the rate-distortion results as a supplementary plug-in denoiser on the ground with no negative effect on compression. It should be noted that these gains only come from noise-aware training.

4.3.2 Image compression optimised for denoising

More advanced methods to help denoise the input data before entropy coding can be implemented in the encoder part of the network. This will help the entropy coding by reducing the noise when the input image is transformed into a latent representation. The work of [K. L. Cheng et al., 2022] uses a two-branch network during training to infer better learning of the noise statistics in the encoder.

As shown in Figure 4.7, this hyperprior-based network has two different behaviours depending on the training or testing time. During training a pair of ground truth and noisy images is fed to the two branches of the network. Each image is transformed into a latent representation by encoding blocks (residual layers) whose weights are shared between the two branches. The difference is that the noisy image passes through a slightly modified version of the encoder, which has additional denoiser blocks (attention modules) in between the regular encoding blocks. After encoding, only the noisy image is processed by the rest of the network, i.e. the hyperprior, the context model and the decoder.

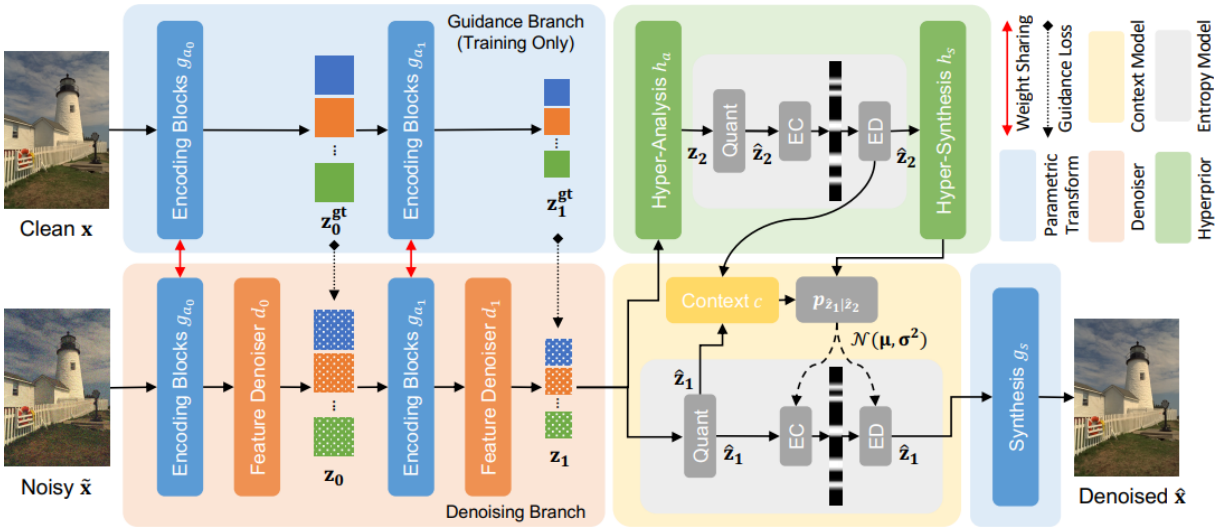


Figure 4.7 – Two-branch design for a joint compression and denoising network trained with pairs of ground truth and noisy images. Source: [K. L. Cheng et al., 2022]

The network is trained in two steps. First, only a pre-training of the compression part (i.e. the guiding branch shared between both branches) was carried out, since the network is essentially a compression network. The parameters are learned over a standard

rate-distortion trade-off. Second a fine-tuning with the denoising branch and the pair of images. This fine-tuning uses an extended version of the rate-distortion loss function with an additional guiding loss:

$$\mathcal{G} = \|z_0 - z_0^{GT}\| + \|z_1 - z_1^{GT}\|$$

These terms are feature maps extracted after the denoiser blocks and compared to the guiding branch features. This multi-scale guidance loss forces the denoiser blocks to adapt to the denoising with an additional constraint in the encoder.

4.4 Joint compression and demosaicking

In most modern digital cameras the demosaicking process is performed before compression. This allows the codec to encode and store a fully processed and restored raw image so that only the decompression operation is required to display the image. However, in a transmission-first approach to the imaging system pipeline (e.g. on board a satellite), the need to compress the CFA before any other process takes place is paramount to save space and time. One of the reasons for this is that demosaicking does not increase the information content, but only creates redundancy in the CFA through interpolation.

Compared to the other joint processing problems (i.e., denoising/demosaicking and denoising/compression), which have extensive literature in both traditional approaches and deep learning methods, the joint demosaicking and compression problem has few occurrences in the literature. Work in the literature that considers both processing is often close to a fully sequential pipeline and does not provide unified end-to-end methods. This is the case in the work of [Gershikov & Porat, 2010], who compresses the CFA pattern of the raw image into four channels RGGB using a DWT transform. After decompression, the four channels are rearranged in the CFA pattern and a demosaicking method is used to recover the full colour image.

Other papers focus on compression-aware demosaicking [Daho et al., 2011]. The CFA is first decomposed from RG_1G_2B into luminance and chrominance $Y_1Y_2C_bC_r$ before DCT transform. Variations exist in the form of a quincunx pattern to extract the green channel

in the luminance channel [Lian et al., 2006]. The DCT coefficients are then encoded and decoded but the demosaicking is done in the DCT domain instead of the RGB domain to save processing steps such as inverse DCT transform and colour transform [Daho et al., 2011]. The trade-off between the two sequential steps and their respective processing error has been analysed in [Lian et al., 2006].

The only joint deep learning method that tackles the problem of demosaicking and compression is the work of [Uhm et al., 2021]. This is a compression-aware demosaicking network whose architecture is presented in Figure 4.8. It is essentially a network that mimics the behaviour of an imaging system pipeline and thus includes demosaicking. However, it is not an agnostic network that only learns on demosaicked data without any information on future compression. They use JPEG as a compression method but since JPEG is not differentiable it cannot be used in the learning of the demosaicking network. To overcome this issue they pre-trained another network that emulates a JPEG compression since the network learns to create JPEG artefact. During training, they use both networks in an end-to-end manner and compute the $L1$ norm as loss functions after each network to evaluate the reconstruction quality. During inference time, only the demosaicking network is used and a regular JPEG codec performs the compression. Their algorithm does not perform both processing but is able to address the issue in a joint manner with a demosaicking network aware of the compression artefact downstream of the pipeline.

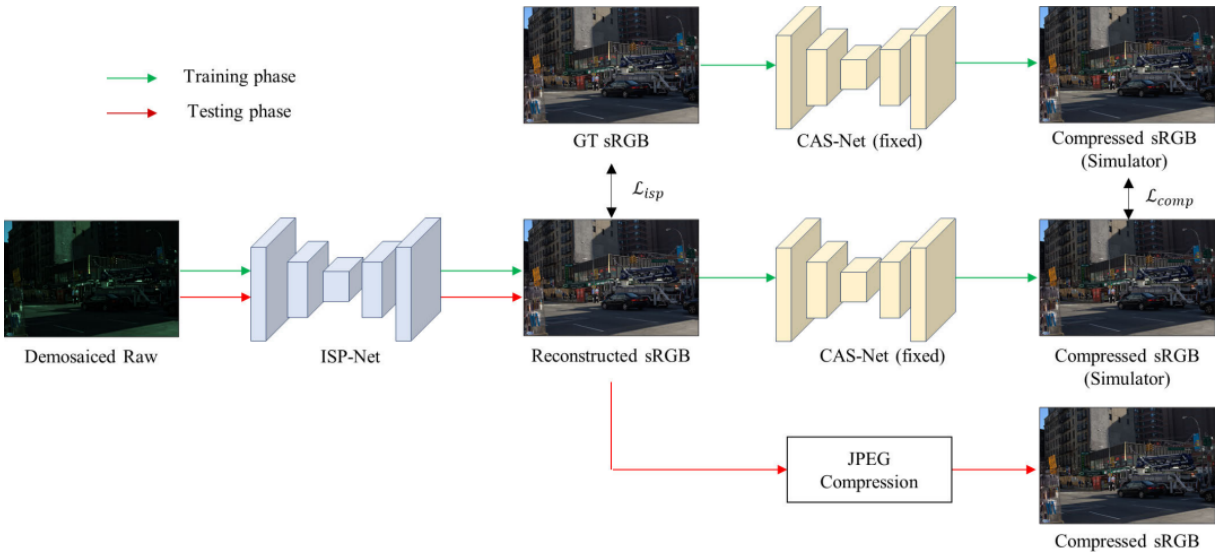


Figure 4.8 – Compression-aware ISP learning. Source: [Uhm et al., 2021]

4.5 Conclusion

In this chapter we have reviewed the literature on inverse problems found in the pipeline of image processing systems from the point of view of joint processing. In addition to compression, which was discussed in the previous chapters of the thesis, we have focused on two other problems: denoising and demosaicking. Merging these processes into a unified method produces better results, both in terms of reconstruction and in terms of saving computational resources.

There has been a great deal of work in the literature on joint processing, with joint denoising and demosaicking on the one hand, and joint denoising and compression on the other. For the last problems (i.e. compression and demosaicking), there is a lack of extensive work, as well as any research on a complete processing pipeline that takes into account these three major steps in any digital camera image processing pipeline. This literature will serve as the basis for Chapter 5, where we aim for an end-to-end optimised network to merge these three steps together.

Chapter 5

Joint compression and processing of satellite images

This chapter is adapted from [Bacchus et al., 2023b, 2023c].

Contents

5.1 Introduction

The main objective of this second chapter of contributions is to extend the work of Chapter 3 with compression to include image processing in our end-to-end framework. In Chapter 3 we focus only on RGB satellite image compression and achieve improved performance while taking into account many of the specific characteristics of satellite imagery from the nature of these images to the use case on board the satellite. However, we deliberately omitted one inherent feature of satellite imagery: onboard acquisition. Our assumption was to consider data that had already been demosaicked and denoised, but these two processes are carried out on the ground. This is because the limited processing capabilities of the hardware do not allow such tasks to be carried out on board satellites.

In this thesis, we consider the cameras of a satellite with high spatial resolution at the cost of lower spectral resolution. Single sensor colour cameras with integrated colour filter arrays (CFAs) are often used to capture the raw image, where each pixel measures the intensity of only one colour band. Efficient algorithms must therefore be developed to compress these remote sensing images before transmission to the ground. Currently, raw data are compressed and transmitted after a KLT transformation. On the ground, decompressed data are demosaicked and denoised. This differs from the usual image acquisition pipeline, which consists of demosaicking the raw data to reconstruct the colour image, which is then denoised and compressed. Denoising in particular is worth doing before compression, as noise is difficult to compress due to the high entropy it induces.

We want to take advantage of the increased hardware capabilities embedded in modern satellites and the improvements brought by neural networks to design a complete imaging processing pipeline from the satellite to the ground centre. To do this, we start with the network architecture design in Chapter 3 and develop a joint compression and demosaicking network. The purpose is twofold: exploit raw image correlations without first interpolating redundancies with demosaicking, and to reduce the complexity of the network by merging several processes. We are improving this architecture with a guiding branch during training for the decoder part of the network to impose a closer feature representation and help with the inverse problem reconstruction. We then want to push our network further by incorporating denoising. Again, the benefit is twofold: it removes the need for any a priori knowledge of the noise with a data-driven method, and it of-

fers us another opportunity to improve high-frequency detail reconstruction by helping the network to discriminate between noise and detail. Finally, we address the issue of bit rate targeting, which is central to codecs. Compression networks often operate as a single rate-distortion model, and some incorporate the possibility of a variable bit rate. However, this type of compression method lacks some form of quality or bit rate constant control, which is crucial for real-world applications.

5.2 Joint compression and demosaicking

This first section focuses on explaining how we address the optimisation of the joint compression and demosaicking problem. The approach we have taken is to not overload the encoder part on board the satellite and leave the reconstruction process, the demosaicking, to the decoder part of the network on the ground. This keeps the compression network light on board, where hardware is limited, but still offers the advantage of joint processing of the raw data. We first give a proof of concept of the benefit of this joint processing network with a straightforward adaptation, and then improve it with a control branch in the decoder to help with demosaicking reconstruction.

5.2.1 Implementation setup

The training procedure shares lots of similarities with that of Chapter 3.1.2. We will focus here on the particularities needed for joint processing. Our whole dataset consists of 300 12-bit RGB images of size 2000 x 2000 with a 50 cm geometric resolution around Lyon and its surroundings. We create 300 raw images filtered with a Bayer colour filter array (CFA) from the ground truth to simulate a raw images dataset. We use a **GRBG** pattern as a CFA filter with a red pixel in the right position, a blue pixel in the bottom left position and green pixels elsewhere in a 2 x 2 square area.

In our supervised method, we work with pairs $(\mathbf{I}, \mathbf{I_B})$ of ground truth RGB noiseless images and their Bayer filtered counterparts. We then separate these pairs of images to create a training and testing dataset consisting of 286 and 14 images respectively. The batch size is set to 8 and at each epoch batch size images are randomly cropped to a 256 x 256 size with data augmentation to increase the variability of the training dataset.

The literature on joint compression and demosaicking is not as extensive as that on joint denoising and demosaicking or joint denoising and compression. To compare our results in the rest of this section, we decide to use several sequential baselines to highlight the gain of joint processing and a joint alternative:

1. Sequential baseline: Malvar demosaicking [Malvar et al., 2004] with our general compression network from Chapter 3
2. Sequential baseline: Malvar demosaicking [Malvar et al., 2004] with JPEG 2000 for the compression
3. Sequential baseline closer to Airbus process: Hamilton-Adams demosaicking [Buades et al., 2011b] with JPEG 2000 for the compression
4. Joint baseline: Oliveira's compression network [Alves de Oliveira et al., 2020, 2021] retrained on raw Bayer images from our dataset.

JPEG 2000 is used instead of the CCSDS 122.0-B lossy compression standard [CCSDS, 2017], as the latter is a slightly degraded version of the former in terms of rate-distortion performance. Malvar demosaicking performance holds up well against modern techniques given its simplicity at a fraction of the cost of specialised networks [Kwan et al., 2019].

5.2.2 Demosaicking capability of our compression network

5.2.2.1 Description of the approach

By construction, joint processing produces better results than separate processing, since separate processing is a particular case of joint processing as seen in Chapter 4. Our proposal is therefore to carry out a global joint processing by explicitly exploiting the statistical dependencies in the raw data. There are 2 approaches to separate processing (also known as sequential processing). Either demosaicking before compression: this process is sub-optimal from a complexity point of view because it increases the size of the data only to have to reduce it afterwards in the compression stage. Another approach is to decorrelate each input and apply independent compression to each channel. However, this process is sub-optimal because only the correlation and not the statistical dependency of the data is exploited. Moreover, decorrelation is generally achieved using a fixed transform and is therefore not adapted to the data.

We present our global architecture that performs joint compression and demosaicking in Figure 5.1. We essentially use our autoencoder architecture with all the improvements of Chapter 3 except for the residual compression, which requires a specialised compression network. The reason for this is that the input and output of the network must have the same dimension and representation in order to compute the residual image from the reconstructed image.

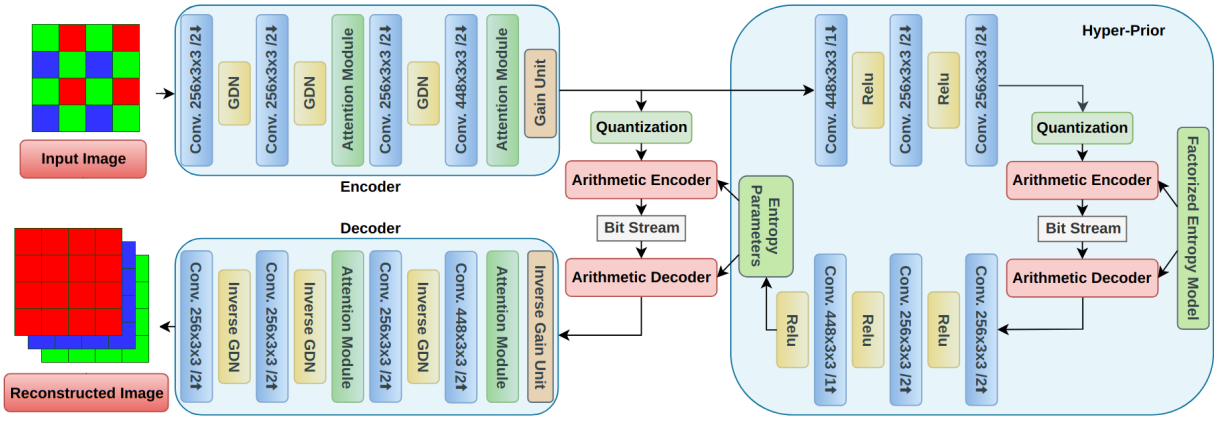


Figure 5.1 – Joint compression-demosaicking architecture with raw data as input.

The hyperprior architecture [Ballé et al., 2018] consists of two autoencoder networks, trained on a pair of ground truth and Bayer filter images (GRBG pattern), with the ground truth as the reference for the loss computation and the raw data as the input to the network. The first autoencoder produces a latent representation y of the input raw data $\mathbf{I_B}$. Standard compression operations such as quantisation and entropy coding are applied to this latent representation to produce a bitstream which is then decoded by the entropy decoder as $\hat{y}$. The decoder reconstructs the signal $\hat{\mathbf{I}}$ with inverse transforms, which is now an RGB image that we want to be close to the ground truth $\mathbf{I}$). The other autoencoder, the hyperprior, models the parameters of the latent representation distribution to improve the entropy model. This shared entropy model is adapted to the characteristics of this input data as the entropy parameters are re-estimated at each input.

Our joint network retains the same complexity as the compression algorithm presented in Chapter 3. Adding processing in the compression network without adding layers improves the network capacity within the encoder part of the network, which means more processing capabilities without increasing complexity. In fact, the total amount of input

data has shrunk from RGB to raw, but the network has stayed the same to compensate for the additional processing required. We approach the issue of complexity from a different perspective. Instead of reducing complexity to maintain equivalent performance, both in terms of the amount of processing and rate-distortion, we keep the complexity constant but with more processing and better rate-distortion performance.

More precisely, we keep all the layer optimisations, additional layers and enhancements from Chapter 3 to achieve a high bit rate and good reconstruction quality. Variable bit rates and tuning of the number of filters and kernel size are also retained for reduced complexity. Concerning the learning of the parameters, we use the rate-distortion-perception trade-off designed with our VGG perceptual loss in our joint compression and demosaicking network:

$$\mathcal{L} = \lambda_1 L_1(x, \hat{x}) + \lambda_2 L_2(x, \hat{x}) + \alpha R(\hat{y}) \tag{5.1}$$

$$L_1(x, \hat{x}) = \lambda_a D(x, \hat{x}); L_2(x, \hat{x}) = \lambda_b P(x, \hat{x}) \tag{5.2}$$

Using D and P, the distortion and perceptual metrics respectively, the MSE and our perceptual loss function based on VGG [Simonyan & Zisserman, 2014] to extract structure within our features and help the network distinguish more structure in the image for better high-frequency reconstruction:

$$P(x, \hat{x}) = \frac{1}{nm}(VGG_{0:2}(x, \hat{x})^2 + VGG_{0:4}(x, \hat{x})^2) \tag{5.3}$$

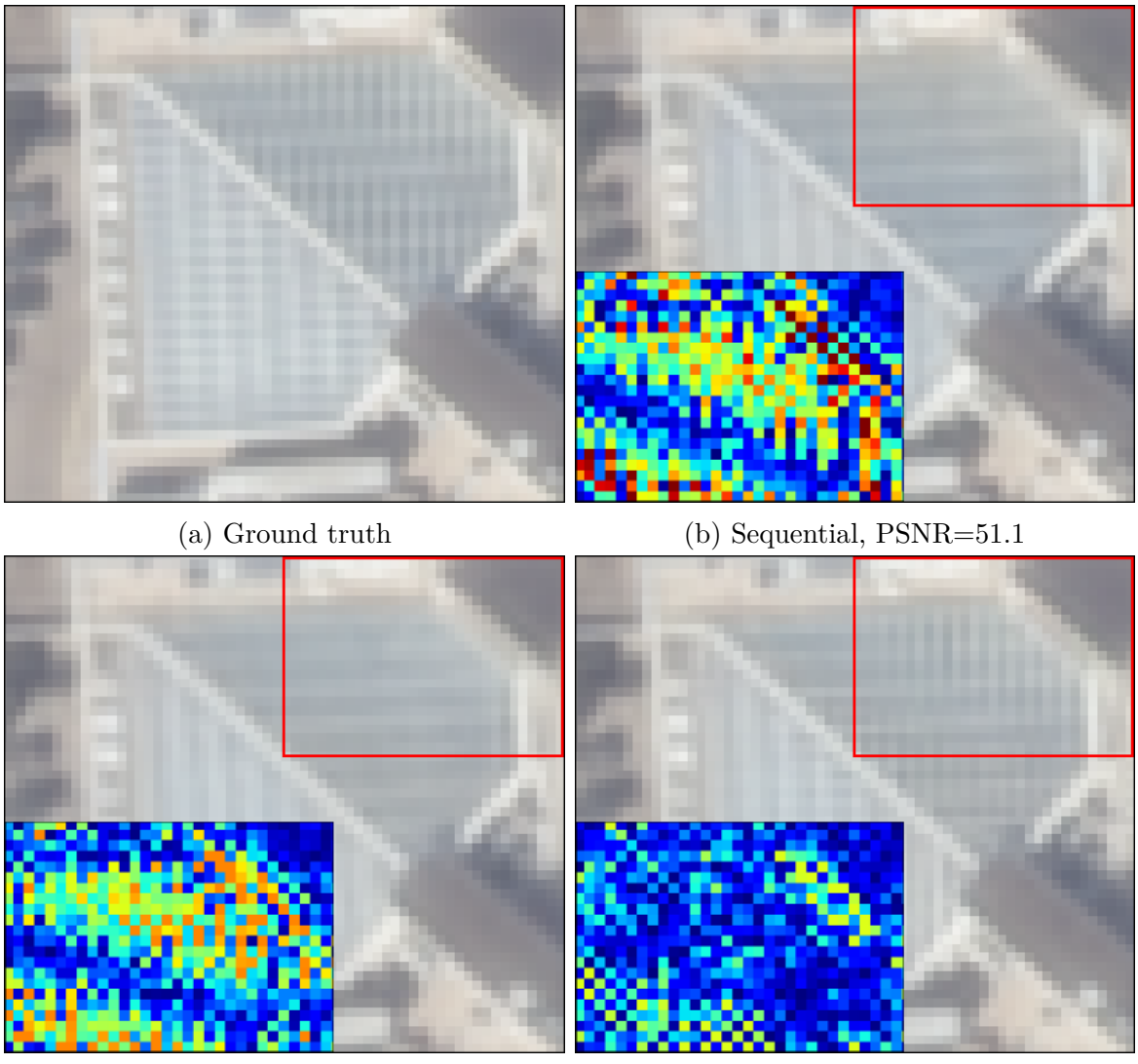
Both λ_a and λ_b are set to $\lambda_a = 2.6 * 10^6$ and $\lambda_b = 10^4$ so that the distortion and perceptual metrics are of the same order of magnitude when the network has converged, α is set to 0.6 to target a medium-high bit rate range around 2 bits per pixel. Finally, λ_1 and λ_2 are automatically tuned using a multi-objective balancing strategy based on previous loss measures for the distortion and perceptual metrics:

$$\lambda_k = K.\frac{exp(\frac{w_k(t-1)}{T})}{\sum_i exp(\frac{w_i(t-1)}{T})}, w_k = \frac{L_k(t-1)}{L_k(t-2)} \tag{5.4}$$

Each loss L_k is linked to its corresponding λ_k, where T controls the proximity of different values of λ_k.

5.2.2.2 Qualitative and quantitative results

Figure 5.2 shows the visual results obtained with different methods compared to the ground truth for a 50cm geometric resolution satellite image of an urban landscape. The ground truth image is compressed at 2bpp with the reference baseline JPEG 2000 with Malvar demosaicking [Malvar et al., 2004], the joint network with multi-loss balancing, and the joint network with MSE loss only.



(a) Ground truth (b) Sequential, PSNR=51.1

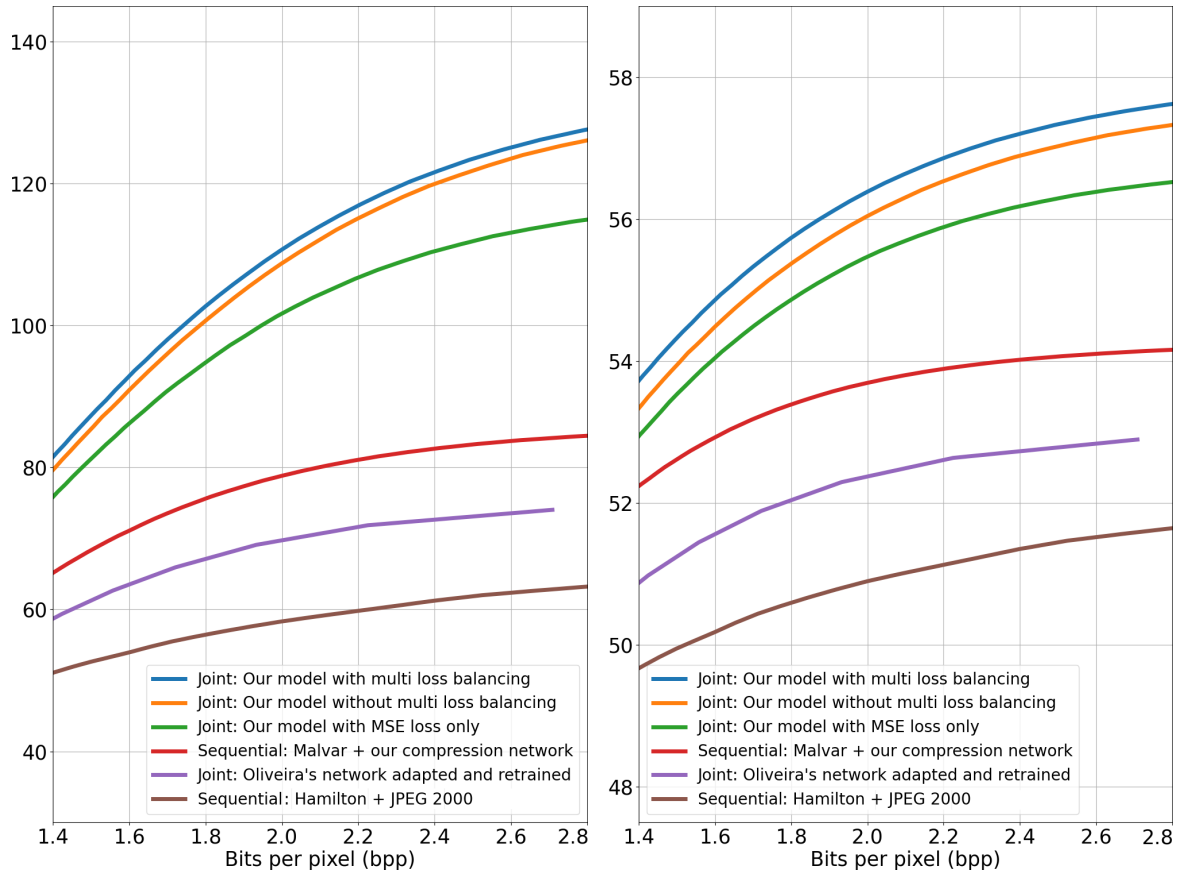
(c) MSE model PSNR=53.6 (d) Our model PSNR=55.5

Figure 5.2 – Visual comparison of compressed images at 2 bpp with the ground truth. An error map shows the relative difference with the ground truth at a pixel level (range value [0;32] from blue to red).

All images are well reconstructed as we aim for a high bit rate. However, in models (b) and (c), high-frequency details such as the striped patterns on the roof of the building have disappeared. Model (d), with the perceptual loss balanced with MSE, comes close to recovering all these details, as the early layers of VGG added more weight to the reconstruction of the structure. When zooming in and analysing the difference in pixel values to the ground truth, we see the impact of a higher PSNR for the deep learning models on the image quality. The pixel difference to the ground truth is much smaller, although this is hard to see at this high bit rate.

We now evaluate the efficiency of the joint processing model with sequential models in Figure 5.3. The joint model achieves a huge bit rate gain at constant quality and outperforms all sequential models. These data-driven models excel at extracting information from irregular data. The joint model can also achieve a reconstruction quality that is infeasible for any sequential model. We can also assess the performance gain that perceptual loss and multi-loss compensation bring to the joint model. Both metrics combined lead to even better performance, especially at high bit rates. This combination of a metric optimised for MSE and one focused on structure extraction reduces the main source of error in our reconstructed images, which comes from high-frequency striped patterns. They have a 1-pixel spatial frequency and disappear due to the blurring generated by the distortion metric, the $L2$ Euclidean norm. Incorporating our perceptual loss allows the network to better adapt to the data characteristics and better preserve high-frequency details during compression with non-complete data, such as raw data.

The multi-loss balancing scheme brings the previous model to a better rate-distortion trade-off over the whole bit rate range. During training, the λ_k parameters adapt to the relative importance given to their respective tasks in previous epochs. This allows the network to escape some local minima as the main λ_k leading the gradient changes over time. We also compare it with other work in the literature on non-hyperspectral satellite image compression [Alves de Oliveira et al., 2020, 2021, 2022]. However, their network is not designed for demosaicking and their dataset differs from ours as they worked with panchromatic data at 70 cm geometric resolution. We are still retraining their networks on our dataset using raw data as input, but the results are completely outside the scope of their work and should be treated with caution.



(a) Distortion measure: SNR (b) Distortion measure: PSNR

Figure 5.3 – RD curves for joint models and sequential baseline for raw satellite images. Demosaicking is performed prior to compression for sequential models.

5.2.3 Improving the inverse problem capability of the network

In the previous section, we demonstrated the efficiency of an inherent compression architecture for joint processing with a simple proof of concept adaptation. We use sub-sampled Bayer data as input to the encoder and impose a full-resolution RGB reconstruction from the decoder output. It gives promising results without any further adjustment of the network. The objective of this section is to propose two new methods using a new architecture and pre-processing in order to to help the network deal with this additional inverse problem.

5.2.3.1 Closer intermediate representation with a guiding branch

The Bayer filtered data was previously considered as a single channel, with each pixel responsible for the intensity of one colour band according to the Bayer pattern (GRBG). However, we want to take full advantage of the supervised learning of our network and the pair $(\mathbf{I}, \mathbf{I_B})$ of ground truth and raw images that we have. The ground truth is only used to compute the distortion and perceptual metric used in our loss, which means that we only evaluate the reconstruction on the fully decompressed input. We want to improve the decoder part of our joint network and enforce a closer representation of the features of the raw input data with the features of the ground truth demosaicked images. To this end, we are designing a learning architecture based on a guiding branch that is effective only during training.

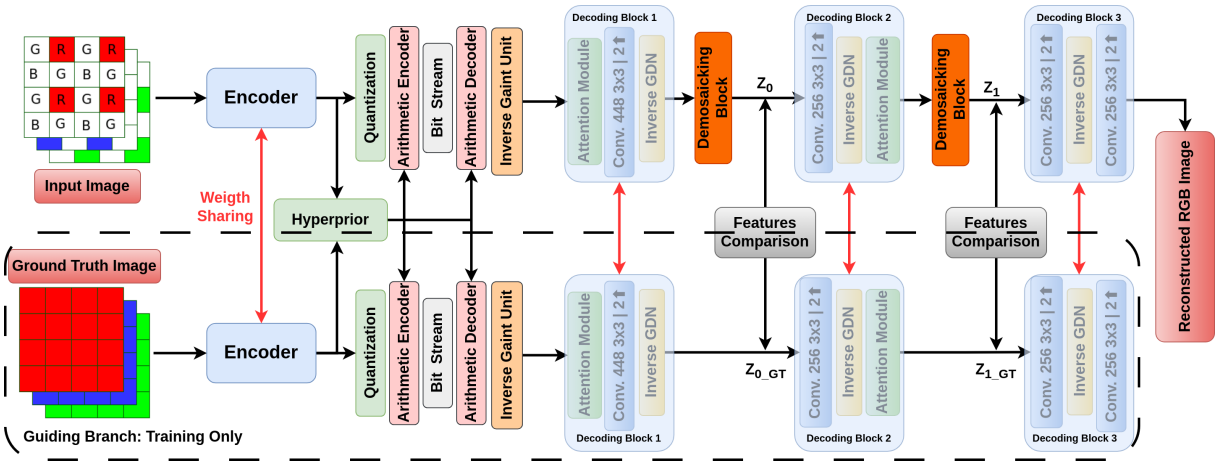


Figure 5.4 – Joint compression-demosaicking architecture with a guiding branch.

The principle as shown in Figure 5.4 is to use a pair of ground truth and degraded images that both pass through the same network. It is used to explicitly supervise the network in image reconstruction for inverse problems [K. L. Cheng et al., 2022]. The ground truth passes through a standard version of the network and the degraded version of the image to an extended version that includes two demosaicking blocks between the decoder blocks. Weights are shared between branches, except for the additional demosaicking blocks. All parameters of the network are learned by backpropagation of the loss so the parameters are learned over the input data. This guiding architecture needs inputs in both branches to have the same dimension as they are using the same network. Previ-

ously in Section 5.2.2, RGB images were sub-sampled to raw data with only one channel. Now we sub-sampled the data in a 3-channel fashion, each one responsible for one colour but only the pixel of the GRBG mask is filled and everything else is set to 0.

We chose to include additional layers in the decoder to assist the demosaicking process for the degraded image. These layers are essentially used to close the gap between the feature maps of the ground truth and the degraded image. The demosaicking layers are a group of several consecutive residual channel attention blocks (RCAB). Residual layers are often found in the demosaicking network [Gharbi et al., 2016; Kokkinos & Lefkimmiatis, 2019].The RCAB layers shown in Figure 5.5 are an extension of the residual layers used in the super-resolution literature [Xing & Egiazarian, 2021; Y. Zhang et al., 2018]. They better extract the channel statistics for improved reconstruction.

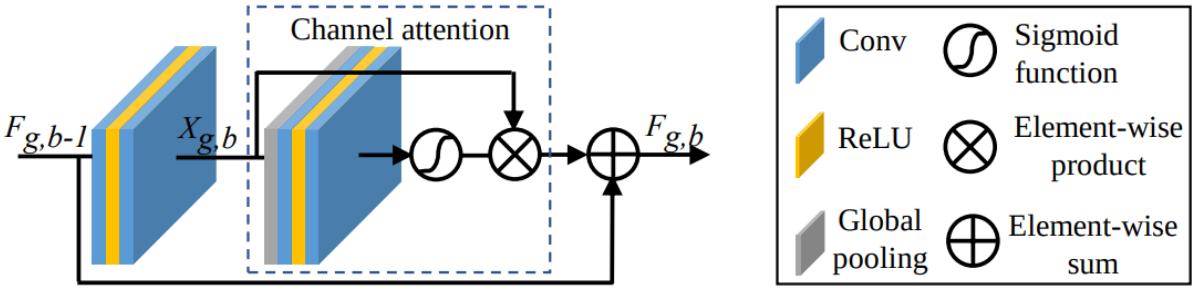


Figure 5.5 – Residual channel attention block (RCAB). Source: [Y. Zhang et al., 2018].

In the decoder part of the network, we extract the feature map from both branches just after the demosaicking block. We create a loss function from these feature maps to add a data anchoring term to force a close representation. As shown in the following equation, we compute the mean of the difference between two sets of pairs and minimise the distance between the demosaicked and the guiding feature:

$$F(x, \hat{x}) = \frac{1}{nm}(|z_0 - z_0^{GT}| + |z_1 - z_1^{GT}|) \tag{5.5}$$

Similar to equation (5.1) with the addition of the perceptual loss in the rate-distortion trade-off. We can extend the learning loss to include this feature representation loss along the rate-distortion-perception trade-off. We continue to use the multi-loss balancing strat-

egy already applied in our previous joint compression and demosaicking network, as the increasing number of hyperparameters makes it even more relevant.

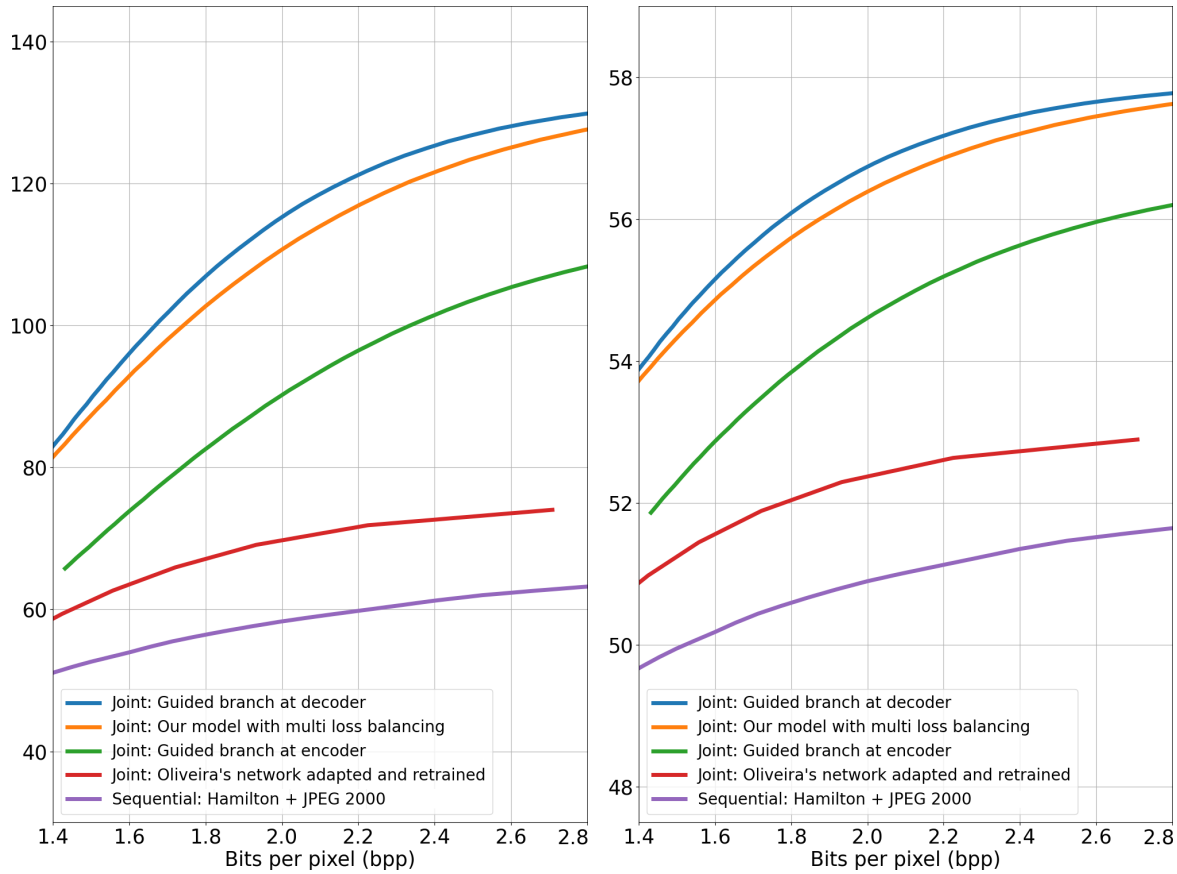
$$\mathcal{L} = \lambda_1 L_1(x, \hat{x}) + \lambda_2 L_2(x, \hat{x}) + \lambda_3 L_3(x, \hat{x}) + \alpha R(\hat{y}) \tag{5.6}$$

$$L_3(x, \hat{x}) = \lambda_c F(x, \hat{x}); \lambda_c = 10^2 \tag{5.7}$$

We now evaluate the efficiency of this guided demosaicking in Figure 5.6. We compare it with the sequential baseline from Airbus and several joint solutions, including our previous joint compression and demosaicking network. We also include a variant of this guided network where the guided branch targets the encoder part of the network. We want to study the effectiveness of the demosaicking process given its occurrence in the network to answer where the guiding is the most beneficial.

If we compare the position of the additional layers and the feature maps needed to compute the loss function, we can see that guiding at the encoder gives poor results. This seems to be a logical consequence, since demosaicking is an inverse problem close to super-resolution, as it tries to interpolate missing data. In our guided network, the decoder part takes the latent representation and decompresses it, but in the meantime, it performs additional computation with the increased network capacity to recover the missing data from the raw input. The role of the encoder is to extract and transform the input into a latent representation that is easier to compress, and this does not combine well with demosaicking. At constant network capacity, this increases the amount of processing done in the encoder for a task that inherently hinders the compression. The guiding branch at the encoder seems more suitable for denoising purposes, as intended in [K. L. Cheng et al., 2022], to remove the noise before compression.

Remark on complexity issue: Our guided network has a consistent rate-distortion gain over the targeted bit range compared to our previous joint model with BD-SNR = 3.30 and BD-RATE = -4.66%. Since the guiding branch is only effective during training and the RCAB layers are in the decoder, which does not run on board the satellite, all the changes made in this section do not affect the inference time in the encoder on board, which means that there is additional rate-distortion for no computational cost on board. Regarding the complexity issue, the encoder variant is also sub-optimal due to the added burden on board the satellite.



(a) Distortion measure: SNR (b) Distortion measure: PSNR

Figure 5.6 – Study of the position of the guiding branch in joint demosaicknig and compression architecture.

5.2.3.2 Support learning with interpolated data

In the previous joint compression and demosaicking network, the raw input was either a single channel where each pixel has its respective RGB colour or a 3-channel input where only the known pixels of each RGB band were filled and all other information was blank. We propose to extend our joint network with a study of data initialisation, to determine whether it is possible to obtain a better reconstruction using pre-processing of controlled complexity. The idea is to use a demosaicking algorithm just before the network in a joint demosaicking and compression pipeline.

To keep this pre-processing as simple as possible, we restrict the test to a few classical demosaicking algorithms. We exclude all deep learning networks, even if they have

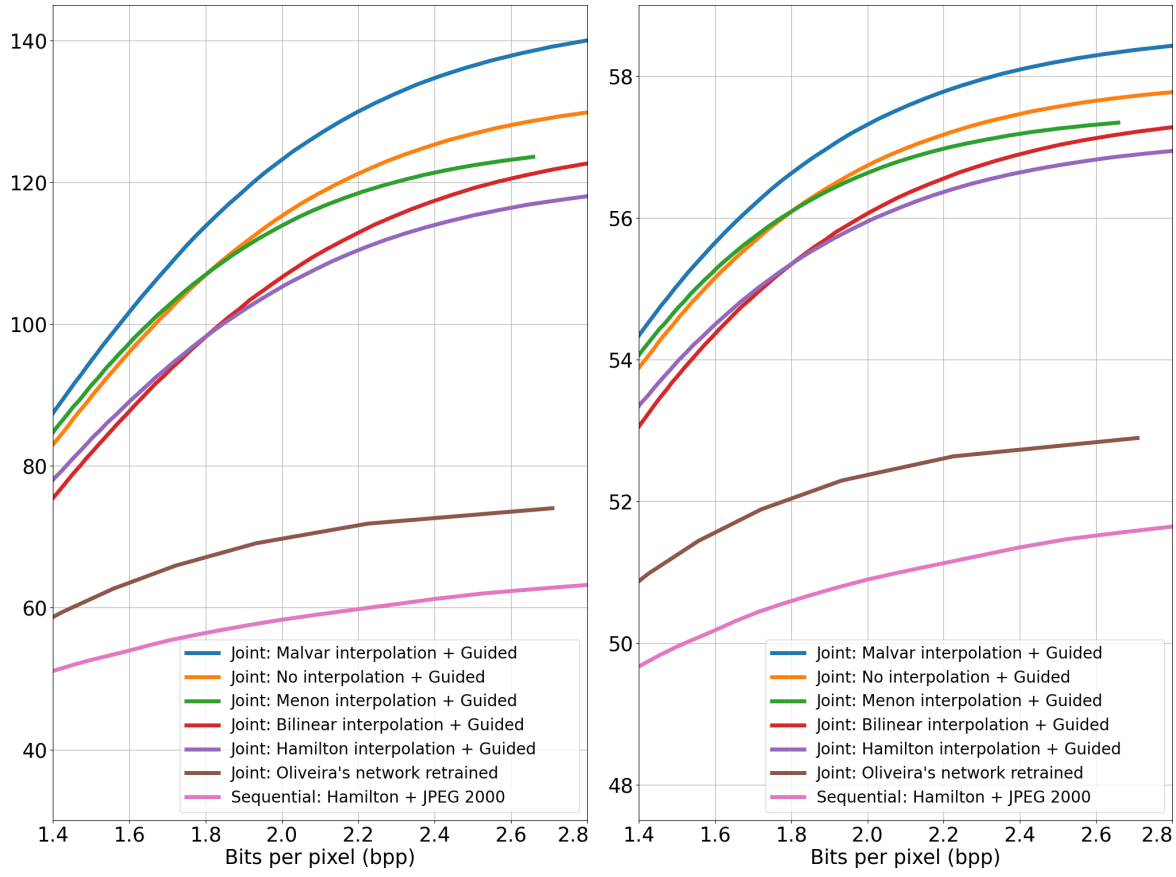
excellent performance, because they are also much more computationally expensive. We selected four demosaicking methods based on their relative reconstruction performance and inference time:

1. Bilinear interpolation [Losson et al., 2010]
2. Malvar demosaicking [Malvar et al., 2004]
3. Menon demosaicking [Menon et al., 2006]
4. Hamilton-Adams demosaicking [Buades et al., 2011b]

We evaluate the efficiency of each demosaicking method as a pre-processing of the input data on our guided architecture. The results in Figure 5.7 show mixed results depending on the methods used. Only Malvar demosaicking is effective compared to our standard guided network with BD-SNR = 7.33 and BD-RATE = -7.85%, all the others give worse results to varying degrees. Menon demosaicking is barely on par with no interpolation, and both bi-linear interpolation and Hamilton demosaicking perform poorly. This may seem surprising as we are feeding the network with better input data that is much closer to the ground truth than a 3-colour channel that is only filled with a quarter (red and blue channel) or half (green channel) of the dimension of the sub-sampled raw data.

However, if we look closely at the demosaicking examples in Figure 5.8 from these algorithms, we can distinguish different visual behaviour, even though the PSNR is close from one algorithm to another. The PSNR metric does not seem to be a very appropriate metric to compare the effectiveness of the different demosaicking algorithms, since the perceptual quality is different. More specifically, Malvar demosaicking has sharper edges compared to Hamilton-Adams demosaicking, which has an overall smoother transition between textures and edges and looks visually closer to the ground truth. Also, colour artefacts seem more present around edges in the bi-linear and Hamilton-Adams methods compared to Malvar. This could be explained by the demosaicking method used by each algorithm. The Hamilton-Adams algorithm [Buades et al., 2011b] uses second order derivatives of the sub-sampled channels to help smooth the gradient-corrected interpolation of the green channel. The Menon method [Menon et al., 2006] uses directional filtering and a posteriori decision. Malvar [Malvar et al., 2004] is a gradient-corrected bi-linear interpolation, which is visually close to bi-linear interpolation, but with an overall better PSNR.

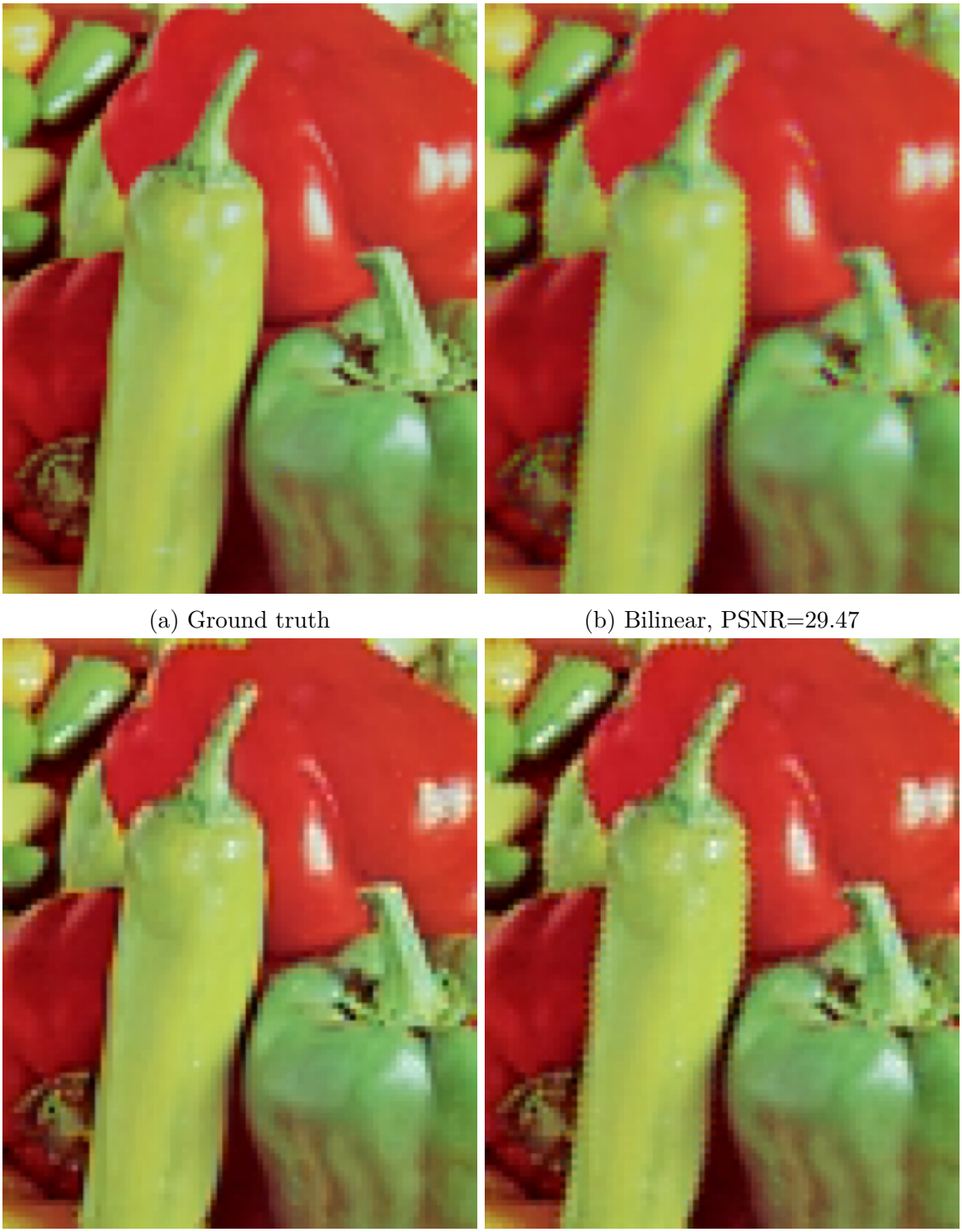
We have seen in Chapter 3 that a deep learning compression network suffers from excessive blur added during compression. The fact that Malvar demosaicking preserves these sharp edges and high frequencies may indicate that it will perform better as a pre-processing algorithm than Hamilton or Menon demosaicking, which produce visually good results but also smooth high frequencies [Jin et al., 2021]. With only a simple single pre-processing of the input data, we can induce better rate-distortion from our joint compression and demosaicking network. This architecture appears to be well adapted to the onboard constraints, with minimal computational overload on board and more computation on the ground where resources are less scarce.



(a) Distortion measure: SNR

(b) Distortion measure: PSNR

Figure 5.7 – Effects of different pre-processing algorithm for raw data in joint demosaicking and compression architecture.



(a) Ground truth (b) Bilinear, PSNR=29.47

(c) Hamilton-Adams, PSNR=29.17 (d) Malvar-He-Cutler, PSNR=29.66

Figure 5.8 – Comparison of different demosaicking methods. Source: [Getreuer, 2011]

5.3 Raw data processing pipeline: Denoising / Compression / Demosaicking

We want to take full advantage of the data-driven behaviour of neural networks to design a complete processing pipeline that can work effectively with raw data. Similar to the compression network in Chapter 3, where we omitted the raw data nature of our images, we have omitted the noise from the sensor acquisition in our joint compression and demosaicking network. Denoising is usually done as post-processing after decompression on the ground, but it can be very useful, especially during encoding [Alves de Oliveira et al., 2022; Brummer & De Vleeschouwer, 2023; K. L. Cheng et al., 2022], to remove noise before compression, as it is costly to compress high entropy information such as noise. It could also be a tool to better distinguish between high-frequency detail and noise for better rate-distortion performance. Finally, with a data-driven method, we remove the need for any noise modelling, since the purpose of the network is to learn it from the data. We propose a single end-to-end network that trains on noisy data to learn the noise model and better reconstruct satellite images.

5.3.1 Extend the network for noisy data

Autoencoders already learn an efficient representation of the input data, which has a denoising effect, since learning the most important features is accompanied by a reduction in the effect of noise on the signal. However, we have seen in both Chapter 3 and Chapter 5 that this is accompanied by a loss of high-frequency detail in the image. Since the noise model of the acquisition system is known, we can work with reliable pairs $(\mathbf{I}, \mathbf{I_{NB}})$ of images corresponding to the ground truth and the noisy Bayer filtered image respectively. The network, as a data-driven algorithm, can extract the specificity of the noisy data. This will increase the ability of the network to not only reconstruct the image from its main features, but also to separate the noise from the high-frequency details. Moreover, this does not require any more computation than supervised training on noisy data, so it has no negative impact on the network complexity and inference time. In this section, we consider the joint demosaicking and compression architecture of Figure 5.4. We work with a pair $(\mathbf{I}, \mathbf{I_{NB}})$ of images and use the ground truth as a reference image to compute the loss terms of the decompressed, denoised and demosaicked images.

5.3.2 Influence of the denoising processing position

The joint approach to denoising with compression and demosaicking also raises the question of the effectiveness of sequential processing. The modular structure of the networks allows several specialised networks to be easily combined, and we want to evaluate the relative gain of a full joint processing pipeline compared to our previous joint demosaicking and compression architecture with a state-of-the-art denoising network.

The choice is large since denoising is a widespread research topic. We choose something similar to what has been done in similar work on joint denoising and compression for satellite images [Alves de Oliveira et al., 2022]. They compare their network with a sequential baseline consisting of their compression network and BRDNet [Tian et al., 2020b]. However, we chose FFDNet [K. Zhang et al., 2018] over BRDNet as our sequential deep learning denoising algorithm. BRDNet achieves a higher PSNR than FFDNet at the cost of much greater computational complexity. It gains about 0.2 dB PSNR on their test dataset, but the inference time varies much more. The hardware in both papers is different, but still, FFDNet is three times faster than DnCNN on the same GPU, while BRDNet is two times slower. For operational purposes, we believe it is more appropriate to use the FFDNet, which combines near state-of-the-art denoising performance with efficient run-time, as it would be a fairer comparison in the context of a hardware-constrained environment.

FFDNet [K. Zhang et al., 2018] is based on the DnCNN denoising architecture [K. Zhang et al., 2017] in terms of the layers used, except that residual connections are removed. The architecture shown in Figure 5.9 shows the down-sampling of the input data with the addition of a noise map.

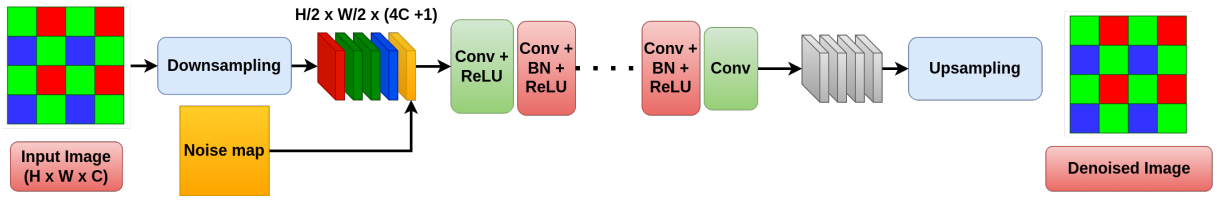


Figure 5.9 – FFDNet architecture [K. Zhang et al., 2018].

This down-sampling eliminates the need to increase the kernel capacity, as convolution is applied to the expanded feature maps. The core of the network is a sequence of D convolution with batch normalisation and ReLU activation function, followed by a final convolution. Up-sampling is then performed to obtain the denoised image. This efficient and effective network is retrained for our data to adapt to the specific noise. It is trained both on the raw data, when it is used before our joint compression and demosaicking network, and on RGB images.

5.3.3 Evaluation

The training procedure is identical to the joint compression and demosaicking training. We will focus here on the specificities required for joint processing. The dataset consists of 300 12-bit RGB images of size 2000 x 2000 with a geometric resolution of 50 cm around Lyon and its surroundings. From the ground truth, we generate 300 noisy raw images filtered with a Bayer CFA (**GRBG** pattern). The noisy images are generated using the photon noise model of the acquisition sensor. We add a noise dependent on the pixel intensity L and the instrumental noise is simulated according to the following equation:

$$\sigma = \sqrt{A + B * L}$$

Parameters A and B are set to obtain images with an SNR of 120 for images with a mean value of 800 over the 2^{12} dynamic range.

We select several baseline methods to compare our full processing pipeline in Figure 5.10. They range from a fully joint model to a partially joint model with sequential processing of one of the three tasks and a fully sequential baseline:

1. Our joint denoising/compression/demosaicking model with a noisy training
2. The PSNR and SNR metric of the noisy validation dataset compared to the ground truth is plotted as a red line on each graph $PSNR = 56.76, SNR = 112.66$.
3. Joint baseline: Our joint **guided** compression and demosaicking network with noiseless training.
4. Joint baseline: Our joint compression and demosaicking network with noiseless training.

5. Partly sequential baseline: Our joint demosaicking and compression network with FFDNet denoising **before** compression on raw data. FFDNet is re-trained on raw data ($\mathbf{I_B}, \mathbf{I_{NB}}$).
6. Partly sequential baseline: Our joint demosaicking and compression network with FFDNet denoising **after** compression. FFDNet is re-trained on our RGB dataset ($\mathbf{I}, \mathbf{I_N}$).
7. Partly sequential baseline: Oliveira's joint compression and denoising network [Alves de Oliveira et al., 2022] retrained on demosaicked [Malvar et al., 2004] data from our dataset.
8. Sequential baseline closer to Airbus process: Hamilton-Adams demosaicking [Buades et al., 2011b] with JPEG 2000 and a non-local Bayes denoising after decompression.

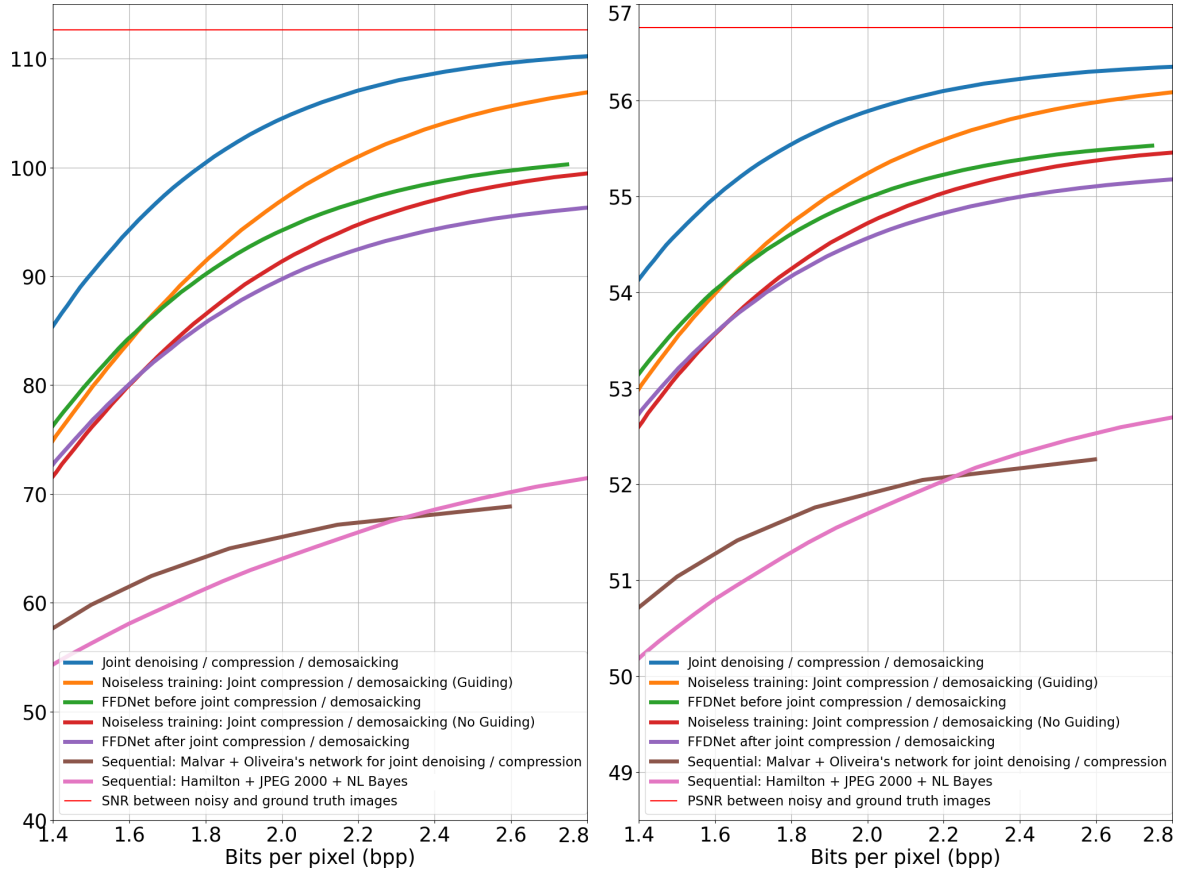
Our full processing pipeline outperforms all other methods by a significant margin over a wide bit range. This is especially true when compared to the sequential method used by Airbus, as we can achieve rate-distortion performance that is not feasible with their method. It is important to note that the noise involved is low ($SNR = 112.66$ for the test dataset), which makes denoising more complex. None of the methods succeeds in returning the image to its original noise level, so in a sense, no denoising has been performed by any of the methods.

However, it should not be forgotten that the processing presented here also includes compression and demosaicking. This processing results in a degradation of the input data compared to the ground truth image, because it involves interpolation of the data with demosaicking and a minimal loss of information with lossy compression. Our method almost reaches the noise level of the noisy image, which means that the addition of compression and demosaicking has added only a small amount of noise compared to the input image.

We have included two results from a network trained on noiseless data but tested on noisy images. We have used our previous joint compression and demosaicking networks trained on noiseless data to evaluate their inherent denoising capability. They perform reasonably well and are on par with the baseline using a joint compression and demosaicking architecture with combined denoising. The autoencoder extraction of the main features make the network inherently good at denoising. However, our full processing pipeline captures the noise information better and therefore removes the noise more ef-

ficiently, resulting in an overall improvement in reconstruction quality with a BD-SNR = 9.91 and BD-RATE = -18.6% compared to the noiseless training of our guided joint demosaicking and compression network.

The addition of the FFDNet denoising as pre or post-processing has a limited effect as networks trained on noiseless data achieve similar performance. However, we can still analyse the effect of the position of the FFDNet. When denoising is done before compression, we observe an increased rate-distortion performance over the whole bit range compared to post-processing denoising. This is not surprising as noise increases the entropy of the image and makes it more difficult to compress. However, this method, which is easily adjustable using a modular network, is also the most computationally expensive, as it requires more processing on board the satellite.



(a) Distortion measure: SNR

(b) Distortion measure: PSNR

Figure 5.10 – RD curves of our full pipeline: denoising, compression and demosaicking.

5.4 Towards constant quality compression

5.4.1 Background

The majority of state-of-the-art end-to-end compression models [Minnen et al., 2018; Minnen & Singh, 2020] are trained over a single rate-distortion point. As seen in Section 3.2.1, this requires more memory to use the different models on the fly and also increases the loading time cost. This is particularly problematic in hardware-constrained environments such as satellites. To solve this problem, we use gain units before quantisation, which act as quality parameters that are added at inference time to allow a variable bit rate for a single model. Although this solves the problem of having a variety of networks on board, it does not help us to target a specific rate-distortion point for our input image. At the same target rate, different input images should be quantised at different levels, corresponding to different gain vectors. We need to find a way to control how the gain units affect the rate-distortion performance for different inputs.

A critical aspect of satellite image compression is the ability to control compression while maintaining quality [Huang, 2011a]. The landscape can vary greatly from image to image, and so can the bit rate for a given quality, from as little as 1 bit per pixel for ocean imagery to more than 4 bits per pixel for city landscapes. We are less concerned with constant bit rate control because we need to reconstruct at a sufficient quality for interpretation. With constant bit rate compression, we would either lose detail on complex images or waste bit rate.

Several papers explore the tuning of gain units with neural networks in the autoencoder architecture [L. Wang et al., 2022; S. Zhang et al., 2022]. The first training is done to learn the parameters of the compression architecture, then it is fixed for the training of the secondary network, which learns to associate a quality parameter for a given quality/bit rate and a given image statistic. The training is performed on the full compression network because the results are used to determine the quality/bit rate of estimation and to compute the loss function of the secondary network. At run time, the latent representation and the target quality/bit rate are fed to this secondary network, which returns an estimate of the gain units required before quantisation and entropy coding. This method gives good results but relies on a lot of processing in the encoder, which we want to avoid in embedded systems.

5.4.2 Method

We want to find a good enough correlation between the quality parameter for a given SNR target and simple statistical tests on the input image. In the paper by [Jiang et al., 2017], the authors use several variables in their decision tree model to estimate the parameters needed to target a particular bit rate. These include continuous variables, such as the decay rate of singular values, or discrete variables, such as the proportion of singular values that retain 95% of the energy. In our case, we select 100 random patches of the same size as the kernel used in our convolution layer (3 x 3) in the input image and create a matrix of these normalised vectors. We then compute the eigenvalues of this matrix and obtain its variance. We also use the ratio of the first two eigenvalues as a metric. Finally, we measure statistics of the image such as the variance.

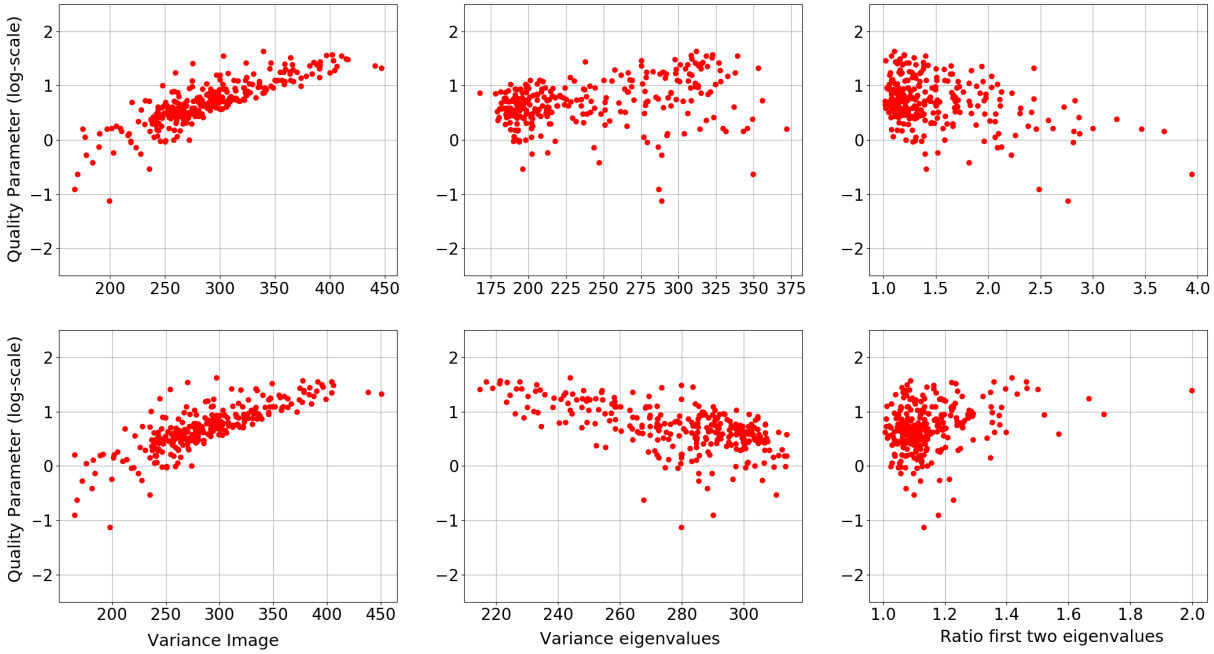


Figure 5.11 – Quality parameter (target SNR=90) as a function of several input metrics: variance of the image, variance of the eigenvalues of a subset of the image, ratio of the first two eigenvalues. The first row is for Bayer filtered images. Second row is for demosaicked images (Malvar).

In Figure 5.11, we plot the results of the quality parameter for an SNR target of 90, given each metric for both Bayer filtered data (guided architecture with raw images) or demosaicked data (guided architecture with interpolated images). The correlation obtained

between the input metric and the quality parameter varies considerably. The ratio of the eigenvalues offers a too loose correlation compared to the other two input metrics. The variance of the eigenvalues does not provide a meaningful correlation for raw data. The best correlation is obtained with the variance of the input image, regardless of whether the image is raw or demosaicked.

Our estimation model consists of using the variance of the training data and computing the convex hull of the point cloud in the form of a step function. This choice is motivated by the need to have strict control over the lowest quality compressed images can have. We want to ensure that the SNR obtained with our estimation is at least the target we impose, rather than approximately the target. In the latter case, there would be a variation in the SNR obtained with a standard regression model, as we can see outliers in the point cloud that will fit poorly. Such an estimation would give a better bit rate average but would sacrifice images that are more difficult to compress (typically urban landscapes). Although reducing the bit rate while maintaining quality is important in our compression scheme, achieving sufficient compression quality is equally important.

We aim for a ratio of total noise to instrumental noise of around 1.2, which, if we ignore the slight denoising due to compression, means that the compression noise is about 3 times lower than the instrumental noise. For satellite imagery, this is usually equivalent to an average of 3 bits per pixel, but this varies with the scene, from over 4 bits per pixel for urban landscapes to less than 1 bit per pixel over the sea. Given the noise model developed earlier, if we aim for a strict ratio of 1.25 for our test dataset ($SNR = 112.66$), then the SNR to aim for is 90. In Table 5.1, for SNRs ranging from 100 to 85, we show the bit per pixel from our estimate compared to the optimal bit rate required to at least meet the target. The relative increase in bit rate ensures that we reach the target level. This ranges from a bit rate increase of more than 30% SNRs greater than 95, to around 10% increase over the optimum for SNRs around 90. This works well for high bit rates, which is the operating range of our compression network, and with a minimal statistical test, we guarantee a compression quality sufficient for reconstruction purposes.

SNR Target	BPP Estimation	BPP Optimal	Delta Rate	Delta Rate %
100	3.082	2.441	0.641	0.263
99	2.99	2.225	0.765	0.344
98	2.889	2.11	0.78	0.37
97	2.842	2.041	0.801	0.392
96	2.804	1.988	0.816	0.41
95	2.77	1.942	0.827	0.426
94	2.284	1.784	0.499	0.28
93	2.008	1.698	0.31	0.183
92	1.895	1.648	0.247	0.15
91	1.815	1.607	0.208	0.129
90	1.754	1.571	0.183	0.116
89	1.701	1.538	0.163	0.106
88	1.656	1.509	0.147	0.097
87	1.618	1.482	0.136	0.092
86	1.579	1.455	0.123	0.085
85	1.546	1.431	0.115	0.08

Table 5.1 – Bits per pixel for a fixed lower bound SNR target with our estimated quality parameters compared to the optimal bit rate achievable.

5.5 Conclusion

This Chapter presents a joint processing model designed for raw satellite images with a GRBG Bayer pattern. With this model, we simultaneously address several problems of satellite image compression. We answer the questions of rate-distortion performance, raw data adaptation, and computational efficiency while taking into account the nature of the images, the hardware-constrained environment, and the need for faithful compression.

First, we take advantage of the data-driven nature of neural networks to match the input to our RGB compression network to the raw data with a simple proof-of-concept network. This joint processing not only provides better rate-distortion performance than other sequential baselines but also increases the efficiency of the network by combining

multiple processes. This allows us to do more processing with the same RGB compression network than before and avoids the need for additional, less powerful demosaicking.

This joint compression and demosaicking is further enhance by adding a guiding branch during training to help the decoder in the inverse demosaicking problem. We induce a closer intermediate representation in the decoder with a loss function based on features of a pair $(\mathbf{I}, \mathbf{I_B})$ of ground truth and raw data with specialised demosaicking blocks in the decoder. This improvement is obtained exclusively at the decoder level and therefore does not incur any cost for the encoder. We then explore initialisation of the input data with minimal pre-processing in the form of simple demosaicking before feeding the network. This further increase the rate-distortion trade-off of our network.

We then extend this approach to include denoising in a full raw data processing pipeline. This brings the method closer to real-world conditions, taking into account both the raw and noisy nature of the data. Compared to noiseless training, this is a significant gain without any drawbacks. Finally, we look at constant quality compression. This last step is essential for all compression methods, and especially for satellite imagery, where we need all images to have a sufficiently high reconstruction quality for analysis. We perform simple and fast statistical tests on the input images to estimate the appropriate quality parameter for the quality target.

The proposed deep learning framework manages to effectively respond to the problem of satellite image compression with improved rate-distortion performance compared to the traditional sequential baseline.

Conclusion

This thesis addresses the problem of image compression in the context of satellite imagery. This problem is approached in terms of a trade-off between rate-distortion performance and the ability of the network to operate in a constrained hardware environment such as a satellite.

We present a neural network designed for satellite image compression and show the success of deep learning in extracting image features for these specific images. To optimise our compression algorithm for embedded systems, we adjust the network parameters (layers, filters, kernels) to reduce the computational cost, improve performance at high bit rates while maintaining satisfactory reconstruction quality, and enable the network to compress images over a wide range of bit rates. To further improve image quality and compress the high-frequency stripe patterns in our satellite images, we incorporate attention modules into our network. As with all networks, the training step is essential, and we improve the reconstruction of high entropy images with contributions to data augmentation and training losses. We add a perceptual loss function to the rate-distortion trade-off to help extract previously blurred areas during training and we jointly optimise this rate-distortion-perceptual trade-off with a multi-objective balancing strategy. Finally, we present a quasi-lossless compression method that focuses on compressing the high frequencies of the residual image. By filtering and thresholding the residuals, we extract structural information from the reconstruction error in our compression network.

This first series of contributions focuses on RGB images but we aim to extend it to raw images. Based on our compression network, we design a comprehensive processing

pipeline from the satellite to the ground centre to perform joint compression denoising and demosaicking on raw data. The result is that more processing can be done for the same network complexity. The architecture of our compression network has been improved with a guiding branch during training in the decoder part, allowing for closer feature representation and demosaicking reconstruction. Denoising is also incorporated to improve high-frequency detail reconstruction by distinguishing between noise and detail. This results in improved rate-distortion performance and network efficiency compared to the sequential baseline, allowing more processing to be done for the same network complexity. The work is also extended to a constant quality compression method, which ensures that compression never falls below a certain quality threshold, ensuring high reconstruction quality while minimising bit rate.

The proposed joint processing approach outperforms the Airbus reference in terms of rate-distortion and visual quality while offering great versatility in the number of tasks that can be performed by a single network. As the processing power of satellites increases, these methods could prove to be new standards for the industry.

Perspectives

We consider that the joint network developed in this thesis opens up interesting research directions. We believe that there are three main areas we should be looking at in order to continue in the direction of a joint compression network for satellite imagery: optimising our network on dedicated hardware, designing a task-aware compression and increasing the capabilities of our joint processing network.

Hardware implementation

In this thesis, we have considered the complexity of our networks from a high-level viewpoint, with the feasibility of joint processing and little consideration of the energy and computational consumption of the method. In fact, all experiments were performed on a grid of GPUs using sub-optimised Python code, which is far from real-life onboard conditions. Quantisation-aware training or post-training quantisation are methods that aim to reduce the memory footprint of a network by quantifying the weight and activation function [Gholami et al., 2022; Roth et al., 2020]. This comes at a marginal cost in

performance, but also a reduction in floating-point operations per second in dedicated hardware such as FPGAs (Field Programmable Gate Arrays), which are optimised to run with 8-bit integer values instead of the standard 32-bit float. This type of implementation optimisation can have a significant positive impact on onboard image processing throughput.

Improving compression

The core processing in our joint network is still the compression of the bottleneck layer. In order to place greater emphasis on the compression network, the learning of processing tasks outside the imaging pipeline should be included in an end-to-end manner [Pu et al., 2014]. Indeed the whole image processing pipeline is only a part of the whole system, as once on the ground additional tasks are performed on the image depending on the user's needs (e.g. classification) [C. Lee et al., 2015]. Introducing some knowledge of this task could reduce the overall rate-distortion trade-off right after the decoder but improve the performance of the subsequent task. This can be done in conjunction with the switch from uniform scalar quantisation to vector quantisation that has been explored in neural networks [Agustsson et al., 2017; Lu et al., 2019]. From information theory, we know that at a high bit rate, the distortion of an optimal scalar quantisation is $1.53dB$ larger than the Shannon lower bound [Wiegand, Schwarz, et al., 2011]. We can therefore explore this possibility to improve compression performance.

Extending joint processing capabilities

Following the success in jointly addressing compression, denoising and demosaicking, we could further increase the capabilities of our network. A typical processing that comes to mind is image segmentation [Wu et al., 2019]. We could compress the images based on their respective semantic maps to adjust the compression and allocate more bit rates on particular objects found in the image [Akbari et al., 2019]. Some sensitive details could even be losslessly coded to preserve their reconstruction quality. This distinction can take the form of an adaptive quantisation of the latent representation depending on the importance assigned by the semantic map. This would be particularly useful for cloudy images or sea images where only a small part of the image needs to be compressed at a high bit rate (e.g. ships). This would reduce the overall bitstream size while maintaining the high reconstruction quality to analyse the details.

BIBLIOGRAPHY

A Sharif, S., Naqvi, R. A., & Biswas, M., (2021), Beyond joint demosaicking and denoising: an image processing pipeline for a pixel-bin image sensor, *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 233–242.

Agustsson, E., Mentzer, F., Tschannen, M., Cavigelli, L., Timofte, R., Benini, L., & Gool, L. V., (2017), Soft-to-hard vector quantization for end-to-end learning compressible representations, *Advances in neural information processing systems*, *30*.

Agustsson, E., Tschannen, M., Mentzer, F., Timofte, R., & Gool, L. V., (2019), Generative adversarial networks for extreme learned image compression, *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Akbari, M., Liang, J., & Han, J., (2019), Dsslic: deep semantic segmentation-based layered image compression, *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2042–2046, https://doi.org/10.1109/ICASSP.2019.8683541

Alves de Oliveira, V., Oberlin, T., Chabert, M., Poulliat, C., Mickael, B., Latry, C., Carlavan, M., Henrot, S., Falzon, F., & Camarero, R., (2020), Simplified entropy model for reduced-complexity end-to-end variational autoencoder with application to on-board satellite image compression, *7th International Workshop on On-Board Payload Data Compression (OBPDC 2020)*, 1–8, https://hal.science/hal-03079863

Alves de Oliveira, V., Chabert, M., Oberlin, T., Poulliat, C., Bruno, M., Latry, C., Carlavan, M., Henrot, S., Falzon, F., & Camarero, R., (2021), Reduced-complexity end-to-end variational autoencoder for on board satellite image compression, *Remote Sensing*, *13 3*, https://doi.org/10.3390/rs13030447

Alves de Oliveira, V., Chabert, M., Oberlin, T., Poulliat, C., Bruno, M., Latry, C., Carlavan, M., Henrot, S., Falzon, F., & Camarero, R., (2022), Satellite image compres-

sion and denoising with neural networks, *IEEE Geoscience and Remote Sensing Letters*, *19*, 1–5, https://doi.org/10.1109/LGRS.2022.3145992

Anscombe, F. J., (1948), The transformation of poisson, binomial and negative-binomial data, *Biometrika*, *35 3/4*, 246–254.

Aulí-Llinàs, F., Marcellin, M. W., Sanchez, V., Serra-Sagristà, J., Bartrina-Rapesta, J., & Blanes, I., (2016), Coding scheme for the transmission of satellite imagery, *2016 Data Compression Conference (DCC)*, 427–436, https://doi.org/10.1109/DCC.2016.29

Bacchus, P., Fraisse, R., Roumy, A., & Guillemot, C., (2022), Quasi lossless satellite image compression, *IGARSS 2022 - IEEE International Geoscience and Remote Sensing Symposium*, 1532–1535, https://doi.org/10.1109/IGARSS46834.2022.9883135

Bacchus, P., Fraisse, R., Guillemot, C., & Roumy, A., (2023a), Filtered Residual Compression for Satellite Images, *IGARSS 2023 - International Geoscience and Remote Sensing Symposium*, 1–5, https://hal.science/hal-04125811

Bacchus, P., Fraisse, R., Guillemot, C., & Roumy, A., (2023b), Full processing pipeline for satellite images: Denoising, Compression, Demosaicking.

Bacchus, P., Fraisse, R., Roumy, A., & Guillemot, C., (2023c), Joint compression and demosaicking for satellite images, *ICASSP 2023 - IEEE International Conference on Acoustics, Speech and Signal Processing*, 1–5, https://doi.org/10.1109/ICASSP49357.2023.10096011

Bacchus, P., Roumy, A., Fraisse, R., & Guillemot, C., (2023d), Compression quasi sans perte d'images satellites par filtrage de residus, *GRETSI 2023 - XXIXeme Colloque Francophone de Traitement du Signal et des Images*, 1–5, https://hal.science/hal-04156801

Ballé, J., Laparra, V., & Simoncelli, E. P., (2016a), Density modeling of images using a generalized normalization transformation, *4th International Conference on Learning Representations, ICLR 2016*.

Ballé, J., Laparra, V., & Simoncelli, E. P., (2016b), End-to-end optimization of nonlinear transform codes for perceptual quality, *2016 Picture Coding Symposium (PCS)*, 1–5.

Ballé, J., Laparra, V., & Simoncelli, E. P., (2017), End-to-end optimized image compression, *International Conference on Learning Representations*, https://openreview.net/forum?id=rJxdQ3jeg

Ballé, J., Minnen, D., Singh, S., Hwang, S. J., & Johnston, N., (2018), Variational image compression with a scale hyperprior, *International Conference on Learning Representations*, https://openreview.net/forum?id=rkcQFMZRb

Bégaint, J., Racapé, F., Feltman, S., & Pushparaja, A., (2020), Compressai: a pytorch library and evaluation platform for end-to-end compression research, *arXiv preprint arXiv:2011.03029*.

Bischof, R., & Kraus, M., (2021), Multi-objective loss balancing for physics-informed deep learning, *arXiv preprint arXiv:2110.09813*.

Bjøntegaard, G., (2001), Calculation of average psnr differences between rd-curves.

Blau, Y., & Michaeli, T., (2018), The perception-distortion tradeoff, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 6228–6237.

Blau, Y., & Michaeli, T., (2019), Rethinking lossy compression: the rate-distortion-perception tradeoff, *International Conference on Machine Learning*, 675–685.

Bross, B., Wang, Y.-K., Ye, Y., Liu, S., Chen, J., Sullivan, G. J., & Ohm, J.-R., (2021), Overview of the versatile video coding (vvc) standard and its applications, *IEEE Transactions on Circuits and Systems for Video Technology*, *31 10*, 3736–3764, https://doi.org/10.1109/TCSVT.2021.3101953

Brummer, B., & De Vleeschouwer, C., (2023), On the importance of denoising when learning to compress images, *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2440–2448.

Buades, A., Coll, B., & Morel, J.-M., (2011a), Non-Local Means Denoising [https://doi.org/10.5201/ipol.2011.bcm_nlm], *Image Processing On Line*, *1*, 208–212.

Buades, A., Coll, B., Morel, J.-M., & Sbert, C., (2011b), Self-similarity Driven Demosaicking [https://doi.org/10.5201/ipol.2011.bcms-ssdd], *Image Processing On Line*, *1*, 51–56.

Buckler, M., Jayasuriya, S., & Sampson, A., (2017), Reconfiguring the imaging pipeline for computer vision, *Proceedings of the IEEE International Conference on Computer Vision*, 975–984.

Carlavan, M., Blanc-Féraud, L., Antonini, M., Thiebaut, C., Latry, C., & Bobichon, Y., (2012), Global rate-distortion optimization of satellite imaging chains, *On-Board Payload Data Compression Workshop*.

CCSDS, (2005), *Image data compression 122.0-b-1*, Consultative Committee for Space Data Systems.

CCSDS, (2017), *Image data compression 122.0-b-2*, Consultative Committee for Space Data Systems.

Chang, S., Yu, B., & Vetterli, M., (2000), Adaptive wavelet thresholding for image denoising and compression, *IEEE Transactions on Image Processing*, *9 9*, 1532–1546, https://doi.org/10.1109/83.862633

Chappelier, V., & Guillemot, C., (2006), Oriented wavelet transform for image compression and denoising, *IEEE Transactions on Image Processing*, *15 10*, 2892–2903, https://doi.org/10.1109/TIP.2006.877526

Chatterjee, P., Joshi, N., Kang, S. B., & Matsushita, Y., (2011), Noise suppression in low-light images through joint denoising and demosaicing, *CVPR 2011*, 321–328, https://doi.org/10.1109/CVPR.2011.5995371

Cheng, Z., Sun, H., Takeuchi, M., & Katto, J., (2018), Deep convolutional autoencoder-based lossy image compression, *2018 Picture Coding Symposium (PCS)*, 253–257, https://doi.org/10.1109/PCS.2018.8456308

Cheng, Z., Sun, H., Takeuchi, M., & Katto, J., (2019), Learning image and video compression through spatial-temporal energy compaction, *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10063–10072.

Chen, T., & Ma, Z., (2020), Variable bitrate image compression with quality scaling factors, *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2163–2167, https://doi.org/10.1109/ICASSP40776.2020.9053885

Cheng, Z., Sun, H., Takeuchi, M., & Katto, J., (2020), Learned image compression with discretized gaussian mixture likelihoods and attention modules, *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 7939–7948.

Cheng, K. L., Xie, Y., & Chen, Q., (2022), Optimizing image compression via joint learning with denoising, *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XIX*, 56–73, https://doi.org/10.1007/978-3-031-19800-7_4

Condat, L., & Mosaddegh, S., (2012), Joint demosaicking and denoising by total variation minimization, *2012 19th IEEE International Conference on Image Processing*, 2781–2784.

Condat, L., (2010), A simple, fast and efficient approach to denoisaicking: joint demosaicking and denoising, *2010 IEEE International Conference on Image Processing*, 905–908, https://doi.org/10.1109/ICIP.2010.5652196

Crawshaw, M., (2020), Multi-task learning with deep neural networks: a survey, *arXiv preprint arXiv:2009.09796*.

Cui, Z., Wang, J., Gao, S., Guo, T., Feng, Y., & Bai, B., (2021), Asymmetric gained deep image compression with continuous rate adaptation, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10532–10541.

Daho, O., Larabi, C., & Mukhopadhyay, J., (2011), A jpeg-like algorithm for compression of single-sensor camera image, *Proc SPIE*, *7876*, https://doi.org/10.1117/12.872416

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L., (2009), Imagenet: a large-scale hierarchical image database, *2009 IEEE conference on computer vision and pattern recognition*, 248–255.

Dony, R., et al., (2001), Karhunen-loeve transform, *The transform and data compression handbook*, *1 1-34*, 29.

Dumas, T., Roumy, A., & Guillemot, C., (2017), Image compression with stochastic winner-take-all auto-encoder, *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1512–1516, https://doi.org/10.1109/ICASSP.2017.7952409

Dumas, T., Roumy, A., & Guillemot, C. M., (2018), Autoencoder based image compression: can the learning be quantization independent?, *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1188–1192.

Gershikov, E., & Porat, M., (2010), Efficient color image compression using demosaicing, *2010 IEEE 34th Annual Computer Software and Applications Conference*, 244–245, https://doi.org/10.1109/COMPSAC.2010.28

Getreuer, P., (2011), Malvar-He-Cutler Linear Image Demosaicking [https://doi.org/10.5201/ipol.2011.g_mhcd], *Image Processing On Line*, *1*, 83–89.

Gharbi, M., Chaurasia, G., Paris, S., & Durand, F., (2016), Deep joint demosaicking and denoising, *ACM Trans. Graph.*, *35 6*, https://doi.org/10.1145/2980179.2982399

Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M. W., & Keutzer, K., (2022), A survey of quantization methods for efficient neural network inference. *In Low-power computer vision* (pp. 291–326), Chapman; Hall/CRC.

Gleyzes, M., Perret, L., & Kubik, P., (2012), Pleiades system architecture and main performances, *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XXXIX-B1*, 537–542, https://doi.org/10.5194/isprsarchives-XXXIX-B1-537-2012

González, R. C., Woods, R. E., & Masters, B. R., (2009), Digital image processing, third edition, *Journal of Biomedical Optics*, *14*, 376.

Gonzalez, M., Preciozzi, J., Muse, P., & Almansa, A., (2018), Joint denoising and decompression using cnn regularization, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y., (2014), Generative adversarial nets, *Advances in neural information processing systems*, *27*.

Goodfellow, I., Bengio, Y., & Courville, A., (2016), *Deep learning* [http://www.deeplearningbook.org], MIT Press.

Guo, T., Wang, J., Cui, Z., Feng, Y., Ge, Y., & Bai, B., (2020), Variable rate image compression with content adaptive optimization, *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 533–537, https://doi.org/10.1109/CVPRW50498.2020.00069

Guo, Y., Chong, Y., Ding, Y., Pan, S., & Gu, X., (2021), Learned hyperspectral compression using a student's t hyperprior, *Remote Sensing*, *13 21*, https://doi.org/10.3390/rs13214390

Hassen, R., Wang, Z., & Salama, M. M. A., (2013), Image sharpness assessment fbased on local phase coherence, *IEEE Transactions on Image Processing*, *22 7*, 2798–2810, https://doi.org/10.1109/TIP.2013.2251643

He, K., Zhang, X., Ren, S., & Sun, J., (2016), Deep residual learning for image recognition, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

He, D., Zheng, Y., Sun, B., Wang, Y., & Qin, H., (2021), Checkerboard context model for efficient learned image compression, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14771–14780.

He, D., Yang, Z., Peng, W., Ma, R., Qin, H., & Wang, Y., (2022), Elic: efficient learned image compression with unevenly grouped space-channel contextual adaptive coding, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5718–5727.

Hirakawa, K., & Parks, T., (2006), Joint demosaicing and denoising, *IEEE Transactions on Image Processing*, *15 8*, 2146–2157, https://doi.org/10.1109/TIP.2006.875241

Hochreiter, S., & Schmidhuber, J., (1997), Long short-term memory, *Neural computation*, *9 8*, 1735–1780.

Hu, Y., Yang, W., & Liu, J., (2020), Coarse-to-fine hyper-prior modeling for learned image compression, *AAAI Conference on Artificial Intelligenc*.

Hu, Y., Yang, W., Ma, Z., & Liu, J., (2021), Learning end-to-end lossy image compression: a benchmark, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44 8*, 4194–4211.

Huang, B., (2011a), *Satellite data compression*, Springer Science & Business Media.

Huang, B., (2011b), *Satellite data compression*, Springer Science & Business Media.

Jiang, X., Le Pendu, M., Farrugia, R. A., & Guillemot, C., (2017), Light field compression with homography-based low-rank approximation, *IEEE Journal of Selected Topics in Signal Processing*, *11 7*, 1132–1145, https://doi.org/10.1109/JSTSP.2017.2747078

Jin, Q., Facciolo, G., & Morel, J.-M., (2020), A review of an old dilemma: demosaicking first, or denoising first?

Jin, Q., Guo, Y., Morel, J.-M., & Facciolo, G., (2021), A Mathematical Analysis and Implementation of Residual Interpolation Demosaicking Algorithms [https://doi.org/10.5201/ipol.2021.358], *Image Processing On Line*, *11*, 234–283.

Johnson, J., Alahi, A., & Fei-Fei, L., (2016), Perceptual losses for real-time style transfer and super-resolution, *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, 694–711.

Johnston, N., Vincent, D., Minnen, D., Covell, M., Singh, S., Chinen, T., Hwang, S. J., Shor, J., & Toderici, G., (2018), Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4385–4393.

Johnston, N., Eban, E., Gordon, A., & Ballé, J., (2019), Computationally efficient neural image compression, *arXiv preprint arXiv:1912.08771*.

Kendall, A., Gal, Y., & Cipolla, R., (2018), Multi-task learning using uncertainty to weigh losses for scene geometry and semantics, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7482–7491.

Khashabi, D., Nowozin, S., Jancsary, J., & Fitzgibbon, A. W., (2014), Joint demosaicing and denoising via learned nonparametric random fields, *IEEE Transactions on Image Processing*, *23 12*, 4968–4981.

Kingma, D. P., & Ba, J., (2014), Adam: a method for stochastic optimization, *arXiv preprint arXiv:1412.6980*.

Klatzer, T., Hammernik, K., Knobelreiter, P., & Pock, T., (2016), Learning joint demosaicing and denoising based on sequential energy minimization, *2016 IEEE International Conference on Computational Photography (ICCP)*, 1–11.

Klopp, J. P., Wang, Y., Chien, S.-Y., & Chen, L.-G., (2018), Learning a code-space predictor by exploiting intra-image-dependencies, *British Machine Vision Conference*.

Kokkinos, F., & Lefkimmiatis, S., (2018), Deep image demosaicking using a cascade of convolutional residual denoising networks, *Proceedings of the European conference on computer vision (ECCV)*, 303–319.

Kokkinos, F., & Lefkimmiatis, S., (2019), Iterative joint image demosaicking and denoising using a residual denoising network, *IEEE Transactions on Image Processing*, *28 8*, 4177–4188.

Kwan, C., Chou, B., & Bell III, J. F., (2019), Comparison of deep learning and conventional demosaicing algorithms for mastcam images, *Electronics*, *8 3*, https://doi.org/10.3390/electronics8030308

Ladune, T., Philippe, P., Hamidouche, W., Zhang, L., & Deforges, O., (2020), Binary probability model for learning based image compression, *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2168–2172.

Lebrun, M., Buades, A., & Morel, J.-M., (2013), Implementation of the "Non-Local Bayes" (NL-Bayes) Image Denoising Algorithm [https://doi.org/10.5201/ipol.2013.16], *Image Processing On Line*, *3*, 1–42.

Lebègue, L., Cazala-Hourcade, E., Languille, F., Artigues, S., & Melet, O., (2020), Co3d, a worldwide one one-meter accuracy dem for 2025, *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLIII-B1-2020*, 299–304, https://doi.org/10.5194/isprs-archives-XLIII-B1-2020-299-2020

Lebrun, M., (2012), An Analysis and Implementation of the BM3D Image Denoising Method [https://doi.org/10.5201/ipol.2012.l-bm3d], *Image Processing On Line*, *2*, 175–213.

Lee, C., Youn, S., Baek, J. Y., & Sagristà, J. S., (2015), Effects of compression on classification performance and discriminant information preservation in remotely sensed data, *Satellite Data Compression, Communications, and Processing XI*, *9501*, 12–20.

Lee, J., Cho, S., Yoon Jeong, S., Kwon, H., Ko, H., Yong Kim, H., & Soo Choi, J., (2019), Extended end-to-end optimized image compression method based on a context-adaptive entropy model, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Lian, N.-X., Chang, L., Zagorodnov, V., & Tan, Y.-P., (2006), Reversing demosaicking and compression in color filter array image processing: performance analysis and modeling, *IEEE Transactions on Image Processing*, *15 11*, 3261–3278.

Liu, H., Chen, T., Shen, Q., Yue, T., & Ma, Z., (2018), Deep image compression via end-to-end learning, *CVPR Workshops*.

Liu, S., Johns, E., & Davison, A. J., (2019), End-to-end multi-task learning with attention, *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 1871–1880.

Liu, J., Lu, G., Hu, Z., & Xu, D., (2020), A unified end-to-end framework for efficient deep image compression, *ArXiv*, *abs/2002.03370*.

Liu, L., Jia, X., Liu, J., & Tian, Q., (2020), Joint demosaicing and denoising with self guidance, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Liu, Y., Chen, H., Chen, Y., Yin, W., & Shen, C., (2021), Generic perceptual loss for modeling structured output dependencies, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5424–5432.

Losson, O., Macaire, L., & Yang, Y., (2010), Comparison of color demosaicing methods. *In Advances in imaging and electron physics* (pp. 173–265, Vol. 162), Elsevier.

Lu, X., Wang, H., Dong, W., Wu, F., Zheng, Z., & Shi, G., (2019), Learning a deep vector quantization network for image compression, *IEEE Access*, *7*, 118815–118825, https://doi.org/10.1109/ACCESS.2019.2934731

Makitalo, M., & Foi, A., (2011), A closed-form approximation of the exact unbiased inverse of the anscombe variance-stabilizing transformation, *IEEE Transactions on Image Processing*, *20 9*, 2697–2698, https://doi.org/10.1109/TIP.2011.2121085

Malvar, H. S., He, L.-w., & Cutler, R., (2004), High-quality linear interpolation for demosaicing of bayer-patterned color images, *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, *3*, iii–485.

Menon, D., Andriani, S., & Calvagno, G., (2006), Demosaicing with directional filtering and a posteriori decision, *IEEE Transactions on Image Processing*, *16 1*, 132–141.

Mentzer, F., Toderici, G., Tschannen, M., & Agustsson, E., (2020), High-fidelity generative image compression, *Proceedings of the 34th International Conference on Neural Information Processing Systems*.

Minnen, D., Ballé, J., & Toderici, G. D., (2018), Joint autoregressive and hierarchical priors for learned image compression, *Advances in neural information processing systems*, *31*.

Minnen, D., & Singh, S., (2020), Channel-wise autoregressive entropy models for learned image compression, *2020 IEEE International Conference on Image Processing (ICIP)*, 3339–3343.

Ohta, Y.-I., Kanade, T., & Sakai, T., (1980), Color information for region segmentation, *Computer graphics and image processing*, *13 3*, 222–241.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ... Chintala, S., (2019), Pytorch: an imperative style, high-performance deep learning library. *In Advances in neural information processing systems 32* (pp. 8024–8035), Curran Associates, Inc., http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Pu, L., Marcellin, M. W., Bilgin, A., & Ashok, A., (2014), Image compression based on task-specific information, *2014 IEEE International Conference on Image Processing (ICIP)*, 4817–4821, https://doi.org/10.1109/ICIP.2014.7025976

Qian, G., Wang, Y., Dong, C., Ren, J. S., Heidrich, W., Ghanem, B., & Gu, J., (2019), Rethinking the pipeline of demosaicing, denoising and super-resolution, *arXiv preprint arXiv:1905.02538*.

Rad, M. S., Bozorgtabar, B., Marti, U.-V., Basler, M., Ekenel, H. K., & Thiran, J.-P., (2019), Srobb: targeted perceptual loss for single image super-resolution, *Proceedings of the IEEE/CVF international conference on computer vision*, 2710–2719.

Ramanath, R., Snyder, W., Yoo, Y., & Drew, M., (2005), Color image processing pipeline, *IEEE Signal Processing Magazine*, *22 1*, 34–43, https://doi.org/10.1109/MSP.2005.1407713

Ratnasingam, S., (2019), Deep camera: a fully convolutional neural network for image signal processing, *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*.

Rippel, O., & Bourdev, L., (2017), Real-time adaptive image compression, *International Conference on Machine Learning*, 2922–2930.

Roth, W., Schindler, G., Zöhrer, M., Pfeifenberger, L., Peharz, R., Tschiatschek, S., Fröning, H., Pernkopf, F., & Ghahramani, Z., (2020), Resource-efficient neural networks for embedded systems, *arXiv preprint arXiv:2001.03048*.

Ruder, S., (2017), An overview of multi-task learning in deep neural networks, *arXiv preprint arXiv:1706.05098*.

Sajjadi, M. S., Schölkopf, B., & Hirsch, M., (2016), Enhancenet: single image super-resolution through automated texture synthesis, *arXiv preprint arXiv:1612.07919*.

Schwartz, E., Giryes, R., & Bronstein, A. M., (2019), Deepisp: toward learning an end-to-end image processing pipeline, *IEEE Transactions on Image Processing*, *28 2*, 912–923, https://doi.org/10.1109/TIP.2018.2872858

Simonyan, K., & Zisserman, A., (2014), Very deep convolutional networks for large-scale image recognition, https://doi.org/10.48550/ARXIV.1409.1556

Skodras, A., Christopoulos, C., & Ebrahimi, T., (2001), The jpeg 2000 still image compression standard, *IEEE Signal Processing Magazine*, *18 5*, 36–58, https://doi.org/10.1109/79.952804

Sullivan, G. J., Ohm, J.-R., Han, W.-J., & Wiegand, T., (2012), Overview of the high efficiency video coding (hevc) standard, *IEEE Transactions on Circuits and Systems for Video Technology*, *22 12*, 1649–1668, https://doi.org/10.1109/TCSVT.2012.2221191

Syu, N.-S., Chen, Y.-S., & Chuang, Y.-Y., (2018), Learning deep convolutional networks for demosaicing, *arXiv preprint arXiv:1802.03769*.

Tan, H., Zeng, X., Lai, S., Liu, Y., & Zhang, M., (2017), Joint demosaicing and denoising of noisy bayer images with admm, *2017 IEEE International Conference on Image Processing (ICIP)*, 2951–2955.

Tan, R., Zhang, K., Zuo, W., & Zhang, L., (2017), Color image demosaicking via deep residual learning, *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, *2 4*, 6.

Tan, D. S., Chen, W.-Y., & Hua, K.-L., (2018), Deepdemosaicking: adaptive image demosaicking via multiple deep fully convolutional networks, *IEEE Transactions on Image Processing*, *27 5*, 2408–2419, https://doi.org/10.1109/TIP.2018.2803341

Theis, L., Shi, W., Cunningham, A., & Huszár, F., (2017), Lossy image compression with compressive autoencoders, *arXiv preprint arXiv:1703.00395*.

Tian, C., Fei, L., Zheng, W., Xu, Y., Zuo, W., & Lin, C.-W., (2020a), Deep learning on image denoising: an overview, *Neural Networks*, *131*, 251–275.

Tian, C., Xu, Y., & Zuo, W., (2020b), Image denoising using deep cnn with batch renormalization, *Neural Networks*, *121*, 461–473.

Toderici, G., O'Malley, S. M., Hwang, S. J., Vincent, D., Minnen, D., Baluja, S., Covell, M., & Sukthankar, R., (2015), Variable rate image compression with recurrent neural networks, *arXiv preprint arXiv:1511.06085*.

Toderici, G., Vincent, D., Johnston, N., Jin Hwang, S., Minnen, D., Shor, J., & Covell, M., (2017), Full resolution image compression with recurrent neural networks, *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 5306–5314.

Uhm, K.-H., Choi, K., Jung, S.-W., & Ko, S.-J., (2021), Image compression-aware deep camera isp network, *IEEE Access*, *9*, 137824–137832, https://doi.org/10.1109/ACCESS.2021.3116702

Wallace, G., (1992), The jpeg still picture compression standard, *IEEE Transactions on Consumer Electronics*, *38 1*, xviii–xxxiv, https://doi.org/10.1109/30.125072

Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Qiao, Y., & Change Loy, C., (2018), Esrgan: enhanced super-resolution generative adversarial networks, *Proceedings of the European conference on computer vision (ECCV) workshops*.

Wang, Z., Gao, B., Wang, P., Gong, X., & Tong, L., (2021), High-quality fast compression algorithm based on fractal-wavelet, *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, 3900–3903, https://doi.org/10.1109/IGARSS47720.2021.9553261

Wang, L., Mao, X., Zhang, S., & Yang, F., (2022), End-to-end quality controllable image compression, *2022 Picture Coding Symposium (PCS)*, 229–233, https://doi.org/10.1109/PCS56426.2022.10018045

Wiegand, T., Schwarz, H., et al., (2011), Source coding: part i of fundamentals of source and video coding, *Foundations and Trends® in Signal Processing*, *4 1–2*, 1–222.

Woo, S., Park, J., Lee, J., & Kweon, I. S., (2018), Cbam: convolutional block attention module, *ECCV*.

Wu, M., Zhang, C., Liu, J., Zhou, L., & Li, X., (2019), Towards accurate high resolution satellite image semantic segmentation, *IEEE Access*, *7*, 55609–55619, https://doi.org/10.1109/ACCESS.2019.2913442

Xing, W., & Egiazarian, K., (2021), End-to-end learning for joint image demosaicing, denoising and super-resolution, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 3507–3516.

Yang, Q., Yan, P., Zhang, Y., Yu, H., Shi, Y., Mou, X., Kalra, M. K., Zhang, Y., Sun, L., & Wang, G., (2018), Low-dose ct image denoising using a generative adversarial network with wasserstein distance and perceptual loss, *IEEE Transactions on Medical Imaging*, *37 6*, 1348–1357, https://doi.org/10.1109/TMI.2018.2827462

Yu, G., Vladimirova, T., & Sweeting, M. N., (2009), Image compression systems on board satellites, *Acta Astronautica*, *64 9*, 988–1005, https://doi.org/https://doi.org/10.1016/j.actaastro.2008.12.006

Zhang, K., Zuo, W., Chen, Y., Meng, D., & Zhang, L., (2017), Beyond a gaussian denoiser: residual learning of deep cnn for image denoising, *IEEE transactions on image processing*, *26 7*, 3142–3155.

Zhang, K., Zuo, W., & Zhang, L., (2018), Ffdnet: toward a fast and flexible solution for cnn-based image denoising, *IEEE Transactions on Image Processing*, *27 9*, 4608–4622.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., & Wang, O., (2018), The unreasonable effectiveness of deep features as a perceptual metric, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 586–595.

Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., & Fu, Y., (2018), Image super-resolution using very deep residual channel attention networks, *Proceedings of the European conference on computer vision (ECCV)*, 286–301.

Zhang, S., Wang, L., Mao, X., Yang, F., & Wan, S., (2022), Rate controllable learned image compression based on rfl model, *2022 IEEE International Conference on Visual Communications and Image Processing (VCIP)*, 1–5, https://doi.org/10.1109/VCIP56404.2022.10008802

Zhou, L., Sun, Z., Wu, X., & Wu, J., (2019), End-to-end optimized image compression with attention mechanism, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

COLLEGE DOCTORAL BRETAGNE
MATHS, TELECOMS
INFORMATIQUE, SIGNAL
SYSTEMES, ELECTRONIQUE

Université de Rennes

Titre : Compression d'images satellitaires par apprentissage profond

Mot clés : Compression d'images, traitement conjoint, dématriçage, apprentissage profond

Résumé : Les images satellitaires ont une grande résolution aujourd'hui grâce à des capteurs performants. Cela se traduit par une utilisation importante de ces images pour tout type d'application. Cet accroissement du volume de données à transmettre jusqu'à la Terre nécessite des méthodes de compression efficaces devant tenir compte des contraintes matérielles existant dans les systèmes embarqués. Nous répondons à ces problématiques par l'emploi de réseaux de neurones profonds. Nous développons d'abord un autoencodeur adapté au point de fonctionnement voulu et aux particularités des images satellitaires. Nous l'améliorons avec l'ajout d'une fonction de coût perceptuel afin d'extraire les détails hautes fréquences de ces images à forte entropie. Dans un deuxième temps, nous incluons d'autres traitements que la compression à notre réseau pour diminuer la complexité. En effet, les images brutes en sortie de capteurs sont des matrices de filtres colorés exigeant un dématriçage pour obtenir une image RVB. Ces images sont par ailleurs bruitées lors de l'acquisition, ce qui complique la tâche de compression. Nous présentons alors un réseau pour traiter conjointement ces opérations lors de la phase de reconstruction tout en codant des images brutes. Nous améliorons notre réseau avec une branche de guidage lors de l'entraînement pour forcer une reconstruction intermédiaire proche lors du décodage. Notre méthode obtient de meilleurs compromis débits-distorsions que les standards satellitaires actuels tout en réduisant la complexité totale de l'ensemble du processus.

Title: Deep learning for satellite image compression

Keywords: Image compression, joint processing, demosaicking, deep learning

Abstract: Today's satellite images have a high resolution, thanks to high-performance sensors. This means that these images are used extensively for all kinds of applications. This increase in the volume of data to be transmitted to Earth requires efficient compression methods that take into account the hardware constraints onboard. We are addressing these issues by using deep neural networks. First, we are developing an autoencoder adapted to the desired operating point and to the specific characteristics of satellite images. We improve it by adding a perceptual loss function to extract high-frequency details from these high-entropy images. Secondly, we include other processing than compression in our network to reduce complexity. The raw images at the sensor output are colour filter arrays requiring demosaicking to obtain an RGB image. These images are also affected by noise during the acquisition process, which hinders the compression process. We therefore present a network for jointly processing these operations in the decoder while encoding raw images. We improve our network with a guidance branch during training to force a close intermediate reconstruction during decoding. Our method achieves better rate-distortion trade-offs than current satellite baselines while reducing the overall complexity.