Multi-scale Cloud Detection in Remote Sensing Images using a Dual Convolutional Neural Network

Markku Luotamo*, Sari Metsämäki and Arto Klami

Abstract—Semantic segmentation by convolutional neural networks (CNN) has advanced the state of the art in pixel-level classification of remote sensing images. However, processing large images typically requires analyzing the image in small patches, and hence features that have large spatial extent still cause challenges in tasks such as cloud masking. To support a wider scale of spatial features while simultaneously reducing computational requirements for large satellite images, we propose an architecture of two cascaded CNN model components successively processing undersampled and full resolution images. The first component distinguishes between patches in the inner cloud area from patches at the cloud's boundary region. For the cloud-ambiguous edge patches requiring further segmentation, the framework then delegates computation to a fine-grained model component. We apply the architecture to a cloud detection dataset of complete Sentinel-2 multispectral images, approximately annotated for minimal false negatives in a land use application. On this specific task and data, we achieve a **16% relative improvement** in pixel accuracy over a CNN baseline based on patching.

I. INTRODUCTION

REMOTE sensing analytics applications, such as classification of land use, vegetation, urban structures or crop type [1], [2], [3], make use of semantic segmentation i.e. pixel-level classification of images in detecting and visualizing shapes of phenomena and objects on aerial and satellite imagery. When ground-level feature segmentation is based on optical satellite imagery and thus ground reflectance, presence of atmospheric clouds or haze in images is inevitable.

There are various types of clouds as to their shape, optical thickness, extent, height etc. A cloud may affect only one individual pixel, may form vast contiguous surfaces or may be fragmented so that it contains small gaps where visibility to the earth is not completely obscured. Shadows cast by clouds are also problematic [4]. In optical remote sensing, the main problem emerging from presence of clouds is that they partly or totally block the view from the satellite sensor to the ground target. If unaccounted for, clouds and haze can result in false interpretation of an image and therefore cause incorrect conclusions from the application's perspective. To identify the usable pixels of an image, automatically generated *cloud masks* [5], [6] that themselves are semantic segmentations, are invariably used for optical satellite image applications interpreting ground-level phenomena. For interpretation within a given area of interest, typically a representation of cloudless optical reflectance is required. Depending on the extent of cloud cover in the available imagery, this may call for one or more images, each filtered by its respective cloud mask. When

several images are used, their cloudless pixels are composed into a cloudless mosaic image.

Early and still often-used pixel-level classification and estimation methods were based on computationally simple engineered features involving band arithmetics, calculated index values, thresholding or decision trees [7]. More recently, machine learning (ML) methods such as the Support Vector Machine (SVM) [8], Markov Random Fields [9], and in particular deep learning and convolutional neural networks (CNN) have been used for cloud detection [10]. Today, models based on state-of-the-art CNN architectures, such as fully convolutional networks (FCN) [11], UNet [12] or their derivatives, originally developed for semantic segmentation of RGB photographs and biomedical images, are increasingly being used also for remote sensing [13], [10], [14], [15].

Multispectral satellite image size runs in a magnitude markedly different from ordinary photographs. For example, a Sentinel-2 multispectral instrument (MSI) image consists of 13 spectral bands in resolutions of 10m (10980x10980 = 121 Mpx) to 60 m (1830x1830 px). With all bands resampled, for example, to the most accurate 10m resolution, an uncompressed image would consume gigabytes of memory. Even though many CNN-based segmentation methods theoretically scale to arbitrary image sizes, and the processing power of accelerator units such as GPUs is continuously increasing, accelerator memory remains a practical limit [16], [17]. The problem becomes all the more pronounced with high-resolution multispectral satellite images; loading even a single complete full-resolution Sentinel-2 image and processing it with a deep CNN requires more memory than what is available on any single GPU, not to speak of a mini-batch of full images. Solutions to this problem usually involve patching [18], [19], [10] or undersampling [13]. Patching partitions the image into small sub-images and analyzes each one separately, which solves the memory issue but results in loss of global information and limits detection of spatially wide features, since information outside the current patch has no influence on the segmentation. In turn, undersampling naturally loses local information and does not provide pixel-level segmentation at the original resolution. To manage either type of information loss and to detect features of large spatial extent, such as atmospheric clouds, we propose a novel practical segmentation architecture that improves retention of both global and local information.

Our architecture consists of two cascaded CNN model components successively processing undersampled and full resolution images, as illustrated in Figure 1. The first *coarse* component is a modified CNN image classifier that receives an *undersampled* input of a complete image and classifies a full set of fixed-size patches of the whole image with a

*M.Luotamo and A.Klami: University of Helsinki, Dept. of Computer Science
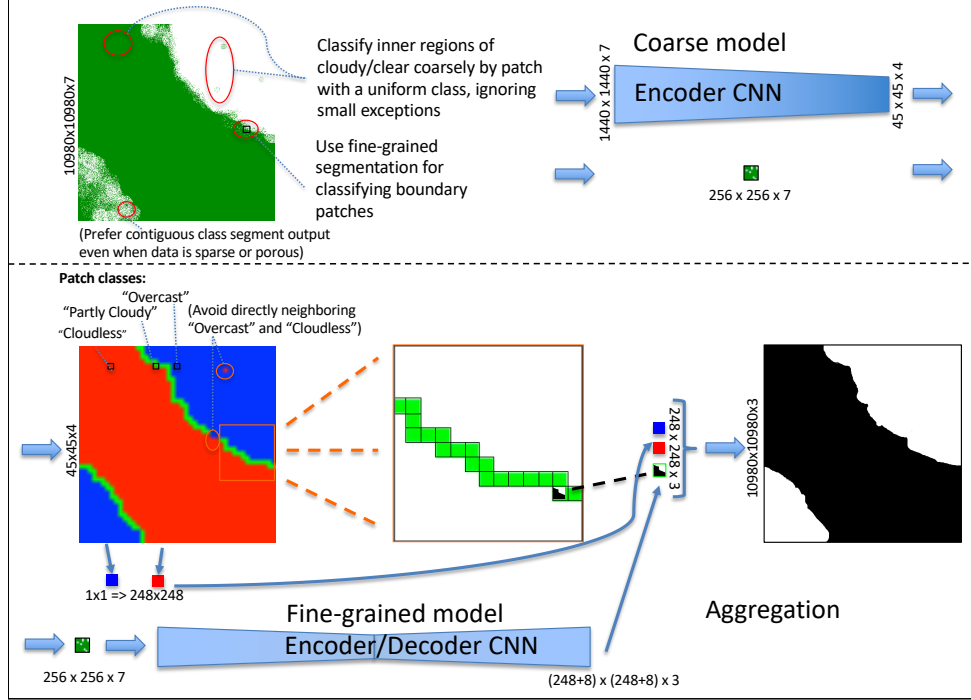Sari Metsämäki: Finnish Environment Institute

Classify inner regions of cloudy/clear coarsely by patch with a uniform class, ignoring small exceptions

Use fine-grained segmentation for classifying boundary patches

(Prefer contiguous class segment output even when data is sparse or porous)

10980x10980x7

Coarse model

Encoder CNN

1440 x 1440 x 7

45 x 45 x 4

256 x 256 x 7

Patch classes:
"Overcast"
"Partly Cloudy"
"Cloudless"
(Avoid directly neighboring "Overcast" and "Cloudless")

45x45x4

1x1 => 248x248

248 x 248 x 3

10980x10980x3

Fine-grained model
Encoder/Decoder CNN

256 x 256 x 7

(248+8) x (248+8) x 3

Aggregation

Fig. 1: Illustration of the main concepts and the overall workflow. **Top:** The *coarse* component analyzes complete MSI images to classify patches as overcast, cloudless or partly cloudy (at the boundary of the cloudy and clear areas). **Bottom:** Subsequently, the *fine-grained* component classifies pixels of "partly cloudy" patches at full resolution.

single class for each patch, in a single pass. To accommodate maximal input resolution with limited memory, we keep the coarse model compact by not including a decoder component, but truncating a standard classifier architecture just before its final encoding layer. The cells of the resulting sparse grid correspond to patches subdivided from the original image. The coarse model assigns one of four classes to each patch. The "Partly Cloudy" class signifies cloud ambiguity and hence a need for pixel-level classification within the patch, whereas the other classes assign a single uniform class to all pixels of the patch (Cloud/Clear/No Data). If a patch is classified as "Partly Cloudy", only then is a second *fine-grained* component used for pixel classification within the patch. This second model is a conventional CNN encoder-decoder that classifies each pixel of its input at the *original full resolution*, using an encoder backbone and a decoder. The two-component coarse-fine architecture enables efficient semantic segmentation of arbitrarily large images while retaining more global and local information than would be possible using exclusively patched or undersampled inputs on a single CNN encoder-decoder.

The requirements of this cascaded architecture are flexible and allow a wide range of choices in assigning different specific CNN architectures as bases for its respective coarse and fine-grained subnetworks. For the fine-grained component, we evaluate a set of recent CNN encoder-decoder architecture variants for semantic segmentation, including PSPNet [20], UNet [12], FPN [21] and Linknet [22]. We vary the encoder part of these CNNs, choosing from different baseline and state-of-the-art encoder/classifier architectures, including VGG-16 [23], ResNet-50 [24], SEResNeXt-50 [25], EfficientNet [26] and Inception-v3 [27]. From this set of encoding classifier architectures, we also select the best-performing ones for evaluation as a basis for the coarse classifier.

We train and evaluate the models on a reference dataset of Sentinel-2 images and cloud masks annotated originally for a land use application [28]. The masks had been manually extended from automated masks to ensure non-cloud-contaminated pixels and to avoid model shortcomings described e.g. in [29]. The annotation guidelines instructed towards *contiguous* cloud regions, as opposed to e.g. porous masks with an abundance of small holes. Small cloud-free areas or pixels had routinely been discarded during annotation. In order to reproduce masks of this nature, we promote high *recall* (see Section V-A on metrics) of cloud pixels within inner cloud regions, allowing a small decrease in precision as a tradeoff. This approach serves many applications better than high precision for a cloud class, which would leave part of the cloudy pixels interpreted as cloud-free. Besides training the model on cloud masks annotated with high recall, we also promote cloud recall and accurate detection of the border region between cloudy and cloudless areas by designing a loss function specifically for these purposes.

To summarize, the main contributions of this work are:

1) A novel semantic segmentation method for high-resolution multi-spectral images using dual cascaded convolutional neural networks. An efficient coarse seg-

mentation retains global patterns and is further focused using a fine-grained model to full resolution at narrow regions of annotation borders, requiring fine-grained processing only locally.

2) A loss function to identify the border segments for fine-grained segmentation, to compensate for the fact that the border areas have a naturally low proportion in the sample distribution, and to explicitly favor contiguous segments for improved emulation of manually dilated cloud masks of the reference data.

3) Demonstration of the approach on cloud segmentation of MSI image data, outperforming both state-of-the-art standalone CNN architectures and well-known baseline cloud detection models in reproducing the high-recall annotations.

II. RELATED WORK

Optical remote sensing analytics has traditionally used cloud detection methods of thresholding, band arithmetics or feature engineering, see for instance work by Ackerman et al. on MODIS data [30], [6], or earlier works by Stowe or Cihlar et al. [31][32]. Since they are computationally efficient to implement, evolved versions of e.g. thresholding decision trees continue to be practical tools today applied to more recent generations of optical satellite imagery for cloud detection [33], [34] as well as other types of pixel-level classification, e.g. snow masks by Metsämäki et al. [35] used in the EU/Copernicus Global snow monitoring service. Foga et al. [36] evaluate several cloud detection algorithms against a Landsat validation mask and take preference for the thresholding-based CFMask due to its global applicability and no need for retraining as opposed to machine learning methods.

Although there were pioneer efforts to apply ML and even neural networks [37] to cloud detection, developments in computing power and increased accuracy of new algorithms have given ground to increased use of various machine learning methods. In particular, advances in computer vision have inspired supervised convolutional neural networks to be applied to semantic segmentation also in the remote sensing and cloud detection context. For example, Mateo et al. [10] showed that their CNN outperformed both a gradient boosting machine and a fully connected multi-layer perceptron, even when the latter two were provided additional features besides the band data.

One of the main challenges in adopting semantic image segmentation advances in remote sensing has been the high dimensionality of the imagery. This still remains a practical challenge despite growing literature, and typical neural network approaches still either train exclusively on small full-resolution patches or on heavily undersampled images. In the context of cloud masking, Shao et al.[18] use a CNN for segmenting inputs of 128x128x10 MSI patches, Yang et al. [19] apply a CNN for 321x321 RGB or grayscale images obtained by patching a downsampled MSI image, and Moharejani et al. [38] used patches of 196x196x4 in combination with QA snow/ice masks. A CNN encoder-decoder of Segal-Rozenheimer et al. [39], inspired by DeepLab [40], uses a module of varying-size dilated convolutions before the feature extraction layer, and eventually trains the network on 256x256 patches.

Besides patching and downsampling, large image size can be addressed by generating superpixels [41] i.e. clusters of similar and adjacent pixels, and then classifying parts of the image only at the superpixel level. Shi et al. [14] assigned a cloud probability to each superpixel's center pixel, based a CNN-classified image patch extracted to center at the same pixel. Xie et al. [15] followed a similar procedure, but using patches of two different resolutions for determining the cloud status, and Liu et al. [42] used CNNs and deep forests on pre-computed cloud superpixels.

Other remote sensing segmentation applications have benefited from the use of CNNs as well, for instance land cover and crop classification [3]. We are inspired by the properties of Fully Convolutional Networks and the UNet architecture and their descendants, as were other authors [38], [19], [13] that used them recently on remote sensing data. However, we apply CNNs in a setting of approximate mask annotations to an MSI image at full and reduced resolutions. The closest work is that of Miyamoto et al. [43], who applied a two-step convolutional network to remote sensing object detection, optimizing for recall. However, their application is far removed from cloud detection and classifies patches instead of pixels. Our approach also has the advantage of modularity; we can use various architectures as building blocks of the cascaded solution and assume the choices that provide the best overall accuracy, as demonstrated in Section V.

Our work also relates more generally to machine learning research on semantic segmentation with various forms of approximate or weak supervision, developed to reduce the high cost of pixel-level annotation of large images. The most common form of weak annotation is to consider bounding boxes surrounding the objects [44], but more elaborate approaches are also being studied. For example, multiple instance learning (MIL) strategies where an annotation signifies that an object is to be found somewhere within the indicated area has been used for segmentation of medical images [45], whereas Shen et al. developed a method that can be trained on crude annotations, each marked somewhere within the object [46]. Pathak et al. [47], in turn, concentrate segmentation around a single maximum-probability pixel. We consider annotations that are supersets of the positive instances and hence formally fit within the MIL framework. However, MIL algorithms are typically developed for scenarios where the positive class covers only a small fraction of the indicated area, whereas in our case, a majority of the pixels annotated as cloudy are indeed cloudy. Hence, the property is better addressed by an improved loss function (Section III-C) instead of dedicated MIL algorithms.

III. METHOD

A. Problem formulation

Given a collection of N high-resolution multispectral images represented as tensors $X_n \in \mathbb{N}^{h \times w \times b}$ and a ground truth[1] binary segmentation $y \in \{0,1\}^{h \times w}$ provided at pixel level for each of the images, the goal is to learn a neural network

---

[1]We use *ground truth* to refer to the target annotation, following machine learning nomenclature. This is not to be confused with in-situ ground-level observations sometimes used as targets; we only use remote sensing data.
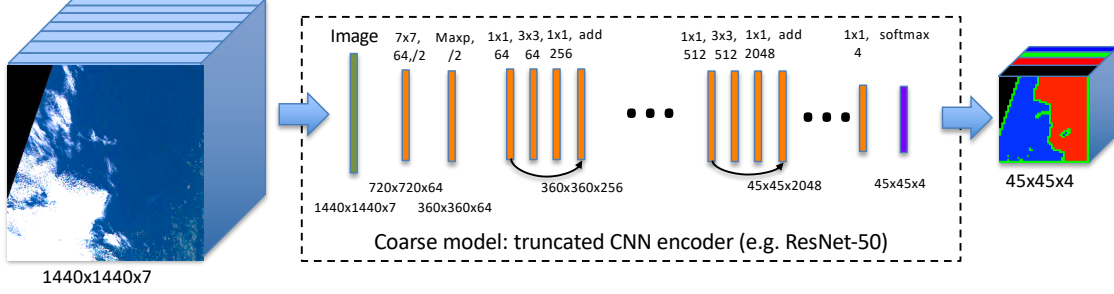
Fig. 2: The coarse component is built based on a CNN encoder, e.g. VGG, Inception or ResNet (illustrated) whose bottleneck layer is dimensioned to a size of e.g. 45x45, segmenting MSI images undersampled to 1440x1440 resolution, into a 45x45 grid of patches each assigned one of four classes.

that can segment future images in a manner that accurately captures the properties of a ground truth segmentation with particular characteristics: In addition to interpreting the spectral composition of a small neighborhood of each pixel to denote clouds, the ground truth annotates a cloud-covered area with a preference for contiguous masks. In this work, we outline a scenario of learning binary cloud masks. However, the technical elements directly generalize to multi-class problems with a moderate number of classes, and are applicable to other domains of large images and contiguous segments.

Within this general description of supervised semantic segmentation problems, we focus on

- Making improved use of both global and local information for increased accuracy.
- Learning from a ground truth that is not accurate at the level of individual pixel. Instead, the data is annotated with coarse contiguous areas of the occlusion class, so that small areas of background within the area are classified as occlusion.
- Good coverage (recall) of the occlusion class; some background classified as occlusion (by clouds) is acceptable, but not vice versa.

In the following, we first explain the overall model architecture in Section III-B and then provide the technical details for a loss function required for addressing the requirements of contiguity and emphasized recall in Section III-C.

B. Model architecture

Deep learning models are mostly trained using hardware acceleration units e.g. GPUs whose available memory per unit remains a practical bottleneck limitation [16][17] for processing large images despite constant improvements in processing speed. Common practice for processing conventional photograph-sized RGB images is to input undersampled images and adapt task objectives to the resulting low resolution of the outputs. For example, full-resolution segmentation is not necessarily required for applications such as identification and tracking of objects. For high-resolution MSI images, however, the relative reduction of segmentation resolution becomes much greater, and significant loss of output resolution is undesirable, if not unacceptable, for many remote sensing applications.

Two main workarounds for processing high-resolution images are to (a) analyze undersampled images [13] as described above, or (b) to analyze isolated smaller patches of the image, looping over multiple patches to process the whole image [10]. Undersampling loses resolution and hence local information, but retains global information better and enables modeling of phenomena of larger spatial extent. Patching, in turn, parallelizes well and retains full resolution, but loses global information and hence has limited ability to model large spatial features. This is both because patching makes it impossible for a model to account for information outside the patch, but also because the geographical extent of CNN features is largely determined by the filter dimensions of the first layer, which is necessarily small when operating at the level of individual pixels.

To alleviate the drawbacks of either approach, we propose a model architecture, illustrated in Fig. 1, that combines coarse analysis of undersampled images with fine-grained analysis for a small number of patches selected by the coarse model. This allows fast and memory-efficient analysis of global features, while retaining a capability for full-resolution segmentation.

We divide the MSI image logically into an e.g. 45×45 grid of constant-sized patches. This split is chosen to yield a manageable patch size for GPU training at full resolution for Sentinel-2 images, but can be easily adjusted to other image dimensions and available accelerator memory. The task of the *coarse* component in our architecture is to provide a classification for each patch, exactly one out of four classes: "Overcast", "Partly Cloudy", "Cloudless", or "No Data" ("No Data" denotes missing data resulting from geospatial transformations from satellite imagery to the orthorectified tiles of Sentinel-2). The coarse model ingests an undersampled input of the complete image, and dimensions are selected so that each patch of the original image corresponds to a cell in the output grid of the coarse model.

Of all patches, only those that were assigned to "Partly Cloudy" are segmented at full resolution with the *fine-grained*

component to pixel-level classes of "Cloud", "Clear", and "No Data". To avoid confusion, we use distinct class names at patch and pixel level (excluding "No Data"). The detailed cloud status, denoted by "Cloud" and "Clear", is available only at the pixel level. At the patch level, "Overcast" refers to a patch for which all pixels are classified as "Cloud". "Cloudless" refers to a patch for which all pixels are "Clear". Finally, "Partly Cloudy" corresponds to a patch having both "Cloud" and "Clear" pixels, i.e. needing more detailed segmentation.

1) Coarse model: The coarse model component (Fig. 2) processes undersampled but spatially complete images. For this, we use the layers of an interchangeable CNN encoder (several are evaluated later), down to the narrowest layer that retains a 2d spatial shape in an image classifier or an autoencoder, i.e. the "bottleneck" layer. We replace the rest of the layers with a dimensionality-reducing 1×1 convolution and a softmax layer to obtain a single classification for each patch of our image on a 45×45 grid of patches.

A core property of the coarse model is its ability to account for features having a large spatial geographical extent. It analyzes images undersampled from 10980 to 1440 in both dimensions (see Section IV and the Supplementary material for details), which means that any convolutional filter covers a roughly 60 times larger spatial area than the corresponding filter would if directly applied at the original full resolution of Sentinel-2 images.

2) Fine-grained model: The fine-grained model component can assume any given pixel-classifying semantic segmentation CNN architecture that is able to operate on 256×256 patches sliced at full resolution from the input image (see Supplement for a representative detailed example on a UNet with a ResNet-50 backbone). The fine-grained model is distinct and separate from the coarse model. In Section V, we present comparative results for several alternatives, eventually selecting FPN architecture with a SEResNeXt-50 encoder backbone. Patch size includes an overlap of 4 pixels with the adjacent patch to minimize patch edge artifacts. Thus, predictions are cropped to a patch size of 248×248. We use a binary cross-entropy loss and a post-processing threshold. Although binary cross entropy is designed to allow multi-class labels per pixel, this choice empirically outperformed categorical cross entropy in our experiments.

C. Coarse model loss function for emphasizing a boundary

For the coarse model component, we want to bring out the "Partly Cloudy" class visible at the cloud boundaries of the derived ground truth (Fig.3), whereas the uniformly masked patches of "Overcast" and "Cloudless" as such provide contiguity to the inner and outer regions of cloud segments. To achieve this, we propose a loss function that can adjust for recall or precision of a class as well as measure and replicate a class' adjacency with other classes against the ground truth.

For all but the very smallest shapes, a raster perimeter drawn around the shape mostly covers a smaller area than the inner or outer area of the shape. This explains the uneven class distribution: the border class ("Partly Cloudy") between fully cloudy ("Overcast") and ("Cloudless") areas is small. The
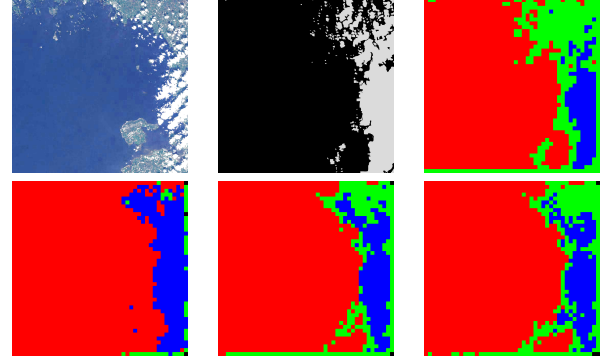

Fig. 3: Coarse model loss function rationale. **Top row** shows the original Sentinel-2 MSI image (left), the binary ground truth mask (middle), and the derived 45x45 ground truth for training the coarse model (right). Here "Partly Cloudy" (green) patches have both "Cloud" and "Clear" pixels, whereas for the other two classes ("Overcast" for blue, "Cloudless" for red) all pixels belong to the same class. **Bottom row** illustrates the predictions of the coarse model for three loss functions. A model trained for unweighted categorical cross entropy (left) completely fails to reproduce the boundary. Using weighted cross entropy helps in recovering the border (middle), and encouraging predictions with correct number of border patches with the adjacency loss (right) captures the border best.

boundary consists of patches that contain both cloud and clear pixels. To improve detection of the boundary, we (a) set weights on weighted cross entropy of each class, and (b) encourage segmentations for which the count of the border patches closely matches the count in ground truth. Intuitively, we would rather allow a small extra amount of pixel-level processing along the borders in between than totally miss a border region between "Overcast" and "Cloudless".

To detect the boundary patches, we optimize for a two-term loss function consisting of a custom *adjacency loss* term, $\mathcal{L}_{\mathrm{adj}}(y, \hat{y})$, and a *weighted cross-entropy* term, $\mathcal{L}_{\mathrm{wce}}(y, \hat{y})$. The overall loss to be minimized is

$$\mathcal{L}(y, \hat{y}) = \gamma \mathcal{L}_{\mathrm{adj}}(y, \hat{y}) + \mathcal{L}_{\mathrm{wce}}(y, \hat{y}), \qquad (1)$$

where $\gamma \in \mathbb{R}^+$ is an adjustable weight between the two terms, y is the ground truth binary indicator tensor and $\hat{y}$ is a predicted probability tensor, both y:s indicating class membership.

$$\mathcal{L}_{\mathrm{wce}}(y, \hat{y}) = -\sum_{k=1}^{C} \alpha_k \sum_{i=1, j=1}^{m,n} [y_{ijk} \log(\hat{y}_{ijk}) + \qquad (2)$$
$$\beta_k (1 - y_{ijk}) \log(1 - \hat{y}_{ijk})] ,$$

The second term, weighted cross entropy (Eq. 2), controls the ratio of each class with a class-specific weights $\alpha_k \in [0, 1]$ and $\beta_k \in [0, \infty[$. C=4 is the number of coarse model classes, $\alpha_k \in [0, 1]$ the importance weight s.t. $\sum_{k=1}^{C} \alpha_k = 1$, and $\beta_k \in [0, \infty[$ is the cross-entropy weight that should be < 1 to reward recall of class k. The values for the hyperparameters used in the experiments are given in Section V.

The first term, *adjacency loss*, is designed so that it is minimized when the number of adjacencies between any two

classes matches between the prediction and the ground truth, in order to emphasize solutions that accurately model class borders. We denote by $S'(y)$ an *adjacency score* that counts the number of adjacencies and define the *adjacency loss* as the squared difference between the prediction and the ground truth

$$\mathcal{L}_{\mathrm{adj}}(y,\hat{y}) = [S'_{\mathrm{d}}(y) - S'_{\mathrm{d}}(\hat{y})]^2. \tag{3}$$

Regularization of segmentation results based on adjacencies has long history in image processing, typically in form of directly penalizing for adjacency of certain classes, either by a Markov random field or specific loss terms such as the one recently proposed by Ganaye et al. [48]. Our formulation is conceptually very different: We do not penalize for adjacency of any classes as such, but instead penalize for a difference in the count of adjacencies between the prediction and the ground truth. This allows finer control of the segmentation result.

Even though we will eventually need a differentiable loss for optimization purposes, we start by defining a *discrete adjacency score* $S_{\mathrm{d}}(y)$ (Eq.4) that counts the total number of adjacently located instances of two pixel classes. For any two classes ($c_{\mathrm{in}} = 2$), the score can be computed using $c_{\mathrm{out}} = 12$ convolutional 2×2 kernel filters K_k, corresponding to the 12 possible adjacency relations (see Figure 4), using

$$S_{\mathrm{d}}(\hat{y}) = \sum_{i=1,j=1,k=1}^{m-1,n-1,c_{\mathrm{out}}} \mathbb{1}[\sigma(\hat{y}) * K_k = 2] \tag{4}$$

Here $\hat{y}$ is a probability tensor, e.g. a two-class subset of a 2D multi-class membership probability mask as output by e.g. a $softmax$ activation in a CNN, of dimension $m \times n \times (c_{\mathrm{in}} = 2)$, σ is a one-hot function turning a real-valued set of vectors into one-hot binary format, and * is the n-dimensional convolution operator here assuming a stride of 1 and no padding. $y_{oh} = \sigma(\hat{y})$ then corresponds to a set of (here, two) mutually exclusive binary 2D pixel masks stacked depth-wise, i.e. a one-hot format of a semantic segmentation, that can be a ground truth or a prediction, of dimension $m \times n \times c_{\mathrm{in}}$, where m and n are width of and height in pixels of a segmentation prediction of an image. Finally, an indicator function $\mathbb{1}$ followed by a summation over $c_{\mathrm{out}} = 12$ filters together count the number of positions that have a value corresponding to a detected adjacency on ($c_{\mathrm{in}} =$)2 bands, i.e. a value of 2. For a normalized, more generalizable metric, we divide $S_{\mathrm{d}}(\hat{y})$ by the amount of pixels and the maximum number $A_{\max}$ of simultaneous adjacency types per filter area (Eq.5). For example, when filter dimensions $m = n = 2$ for a single offending pixel with no appropriate boundary region, $A_{\max} = 4$ (Fig.4).

$$S_{\mathrm{dn}}(\hat{y}) = \frac{S_{\mathrm{d}}(\hat{y})}{mnA_{\max}}. \tag{5}$$

Finally, we convert the discrete adjacency score into a differentiable loss for training the model, by approximating the indicator function $\mathbb{1}$ with the rectified linear unit (ReLU)
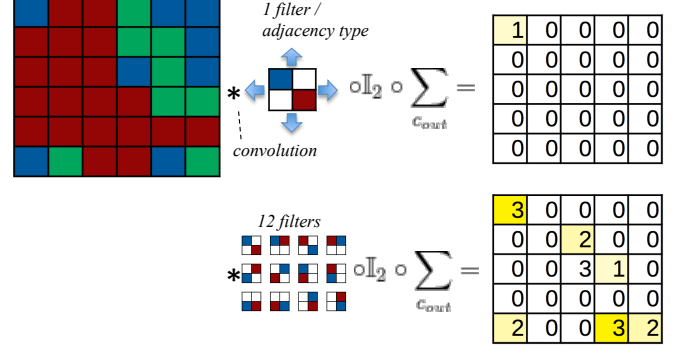

Fig. 4: Adjacency score counts instances of two adjacent pixels on two bands. The score is calculated by moving twelve 2x2 convolution filters for $c_{\mathrm{in}} = 2$ bands over the image, one filter for each adjacency type. Each filter's convolution at two adjacent pixels produces a value of 2 (not shown in the picture, only the count of 2's), hence the indicator function. The highlighted numbers indicate the total count of adjacencies detected by the indicator for all $c_{\mathrm{out}} = 12$ filters.

and a threshold $1 + \epsilon$:

$$S'_{\mathrm{d}}(\hat{y}) = \sum_{i,j,k}^{m-1,n-1,c_{\mathrm{out}}} \mathrm{ReLU}(\sigma'(\hat{y}) * K_k - (1 + \epsilon)) \tag{6}$$

$$s.t. \quad y'_{oh} = \sigma'(\hat{y}) \in \{\{0 + \rho_1, 1 - \rho_2\}^{c_{\mathrm{in}}}\}^{m \times n},$$

$$\sum_{c=1}^{c_{\mathrm{in}}} [\sigma'(\hat{y})]_c \approx 1.$$

Here ϵ and ρ are small residuals, and $\sigma'()$ is a continuous-valued relaxation of one-hot encoding (see Supplement for details). We set ϵ to 0.001 and ρ at floating point precision.

IV. DATA

For training and evaluating the model, we use a dataset of 478 Sentinel-2 (S2) MSI L1C images from years 2016 to 2017, manually selected for a minimal cloud cover and annotated for remaining cloud masks at the Finnish Environment Institute (SYKE). These same masked images have been combined to construct a cloudless mosaic and used as raw data for producing the EU Corine land cover mapping of Finland [28].

A. Annotation

Pixel-level annotation work took several months of work by a team of three annotators, using an initial reference mask produced by the Idepix tool of the SNAP software package by ESA as a basis (masks used were *f_cirrus_sure, f_cirrus_ambiguous, f_brightwhite*). As MSI L1C bands 1-4 and 8-10 were known to contain the most relevant information for cloud masking, they were viewed simultaneously in different false-color combinations to serve as the annotators' additional reference input. As the annotation routine and composition of the team evolved and changed during the course of the annotation work, variability was introduced in the level of
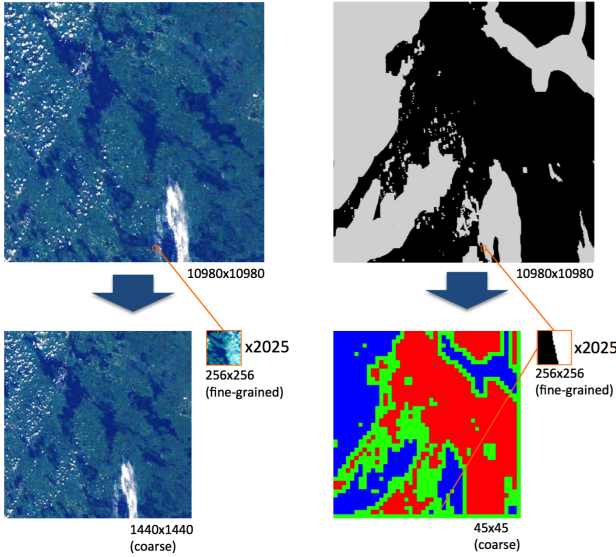
Fig. 5: Illustration of the data pipeline for a single MSI image. **Top left:** Original MSI image data (only visible RGB bands shown here). **Top right:** fine-grained model full-resolution annotations for classes "masked"/ "accepted", **Bottom left:** coarse model data downsampled to 1440x1440 and fine-grained model data as overlapping 256x256 patches. **Bottom right:** for the coarse model, annotations are pooled to a 45x45 grid from full resolution, resulting in classes "Overcast" (blue), "Partly Cloudy" (green) and "Cloudless" (red). For the fine-grained model, overlapping 256x256 patches are extracted from annotations ("No Data" regions omitted from illustration).

detail and features of the resulting manual masks. Spring and winter images contain snow and ice annotated to the same class as clouds since the pixels were unusable from the viewpoint of annotators, considering the targeted land-cover map (there is no permanent snow cover in Finland). Also from the data and algorithm viewpoint, clouds can be a challenge to discern from snow and ice [29].

The annotators were instructed to mask any and all cloud-contaminated areas including partial and sparse occlusion as "Cloud", which in practice results in cloud overestimation. Examining and annotating every pixel in detail would have been an overwhelming and practically infeasible task, even with these relatively ample annotation resources given the size of the dataset, $478 \cdot 10980^2 \approx 58 \cdot 10^9$ multispectral pixels. Since snow was treated as an unusable class just like clouds for the land use application in question, snow-covered areas were likewise annotated as "Cloud". Finally, the cloud edges were automatically extended with a buffer of approximately 10 pixels wide to include mixed pixels in the resulting mask (see Figure 6). Also, "No Data" areas of MSI images were included as part of the "Cloud" class.

B. Data pipeline

In the following, we briefly outline the data processing pipeline used for transforming the original MSI data and the ground truth annotations to form the training and evaluation data sets for the model. Full details of the pipeline are provided in the Supplement.

Using the original two binary classes of the annotations, i.e. "Clear" and "Cloud", we construct three mutually exclusive full-resolution ground truth pixel classes: "Cloud", "Clear" and a reconstructed class for "No Data". "No Data" corresponds to invalid polygonal image regions not covered by the satellite sensors. In the end, we have 10980x10980x7 spectral data with 16-bit depth and a 10980x10980x3 boolean ground truth mask.

Using the outcome of this initial processing step, we derive two separate datasets, one for the coarse and the fine-grained model each. For the coarse model, each MSI image is downsampled to 1440x1440x7, and ground truth masks of 45x45x4 with four classes are derived for patches of the original data so that patches with only "Cloud" pixels are annotated as "Overcast" and patches with only "Clear" pixels as "Cloudless", whereas patches with both are marked as "Partly Cloudy". We augment the coarse model dataset by simple vertical and horizontal flips to increase the data volume, resulting in a total of 1912 images associated with ground truth masks.

For the fine-grained model, we slice each MSI image to dimensions of 45x45x256x256x7. Altogether 478 images yield a total of 975,950 patches of spectral dimensions of 256x256x7 with an overlap of four pixels. Correspondingly, a ground truth mask of 45x45x256x256x3 mask is extracted for the fine-grained model. Due to the contiguous nature of the ground truth, the share of fully "Overcast" or "Cloudless" patches is disproportionately high as noted in III-C. For training the fine-grained model, we downsample the amount of overcast and cloudless, having first divided patches into ten bins by their respective ratio of ground truth "Cloud" pixels. In a natural distribution without weighting, the bins at the extremes of $< 10\%$ and $> 90\%$ of cloudiness ratio have a proportion of more than ten times that of the patches in other bins that are partially cloudy. Naively training on the whole data would bias the model towards outputting patches that have pixel classes of almost uniform "Cloud" or "Clear", yet in our framework, the fine-grained model is only used for patches already known and classified to be only partly cloudy. To remove this bias, we re-weight the ten bins so that the extreme bins have a weight of 16.7% and the remaining eight all have a weight of 8.3%.

C. Model training

The models were trained on 454 of the original 478 MSI images. For the fine-grained model, a proportion of patches from the training images was separated and held out from training into a validation set for estimating optimal hyperparameters. The remaining randomly selected 24 whole images were held out for model testing evaluation, i.e. not used for training or development of the models.

We implemented the proposed architecture with Keras and TensorFlow, using standard stochastic gradient descent with a momentum of 0.9 for training both model components. We initially trained the coarse model from scratch for 131 epochs over 9 hours with weighted cross-entropy $\mathcal{L}_{\text{wce}}$ only, i.e. with $\gamma = 0$. This provides a baseline for $\mathcal{L}_{\text{wce}}$ as such for loss function comparison in Section V-D. For training with the

TABLE I: Comparison of encoder backbones used with a semantic segmentation architecture initially fixed to UNet. Boldface indicates best method for each metric.

Fine encoder	accuracy	precision	recall	f1	iou
EfficientNet	0.5515	0.5112	**0.8308**	0.5769	0.4686
ResNet-50	0.7158	0.7044	0.6422	0.6001	0.4777
VGG16	0.7299	**0.7219**	0.6122	0.6026	0.4817
InceptionV3	0.7336	0.6936	0.6907	**0.6295**	**0.508**
SEResNeXt-50	**0.7345**	0.6898	0.6703	0.6207	0.5006

TABLE II: Comparison of encoder-decoder semantic segmentation architectures, with the encoder backbone fixed to SEResNeXt-50 (see Table I).

Fine segm.	accuracy	precision	recall	f1	iou
UNet	0.7345	0.6898	0.6703	0.6207	0.5006
LinkNet	0.6346	0.5562	**0.736**	0.5807	0.4636
PSPNet	0.6666	0.5841	0.7069	0.5861	0.4675
FPN	**0.737**	**0.7184**	0.6426	**0.6167**	**0.4935**

adjacency loss, we use transfer learning, i.e. assume the weights of the $\mathcal{L}_{\text{wce}}$ baseline as a starting point, activate the $\mathcal{L}_{\text{adj}}$ term with $\gamma = 1$ and train for additional 14 epochs to get the comparison metrics for $\mathcal{L}_{\text{adj}}$. The fine-grained FPN model took 40 epochs over 5 hours to converge from scratch on a random sample of 8000 patches per epoch (500 mini-batches x 16 image patches) out of approx. 1 million patches. We used a threshold of 0.45 for the "Cloud" pixel class, the threshold optimized against a separate validation set.

V. RESULTS AND DISCUSSION

We measure and evaluate the performance of our framework from the points of view of CNN-based semantic segmentation and MSI cloud detection algorithms. As our framework addresses the problem of an imbalanced dataset partly by a custom loss function, we additionally compare loss function alternatives to demonstrate the justification for our choice.

A. Metrics

Metrics used for mutual comparison are accuracy ($\frac{TP+TN}{TP+TN+FP+FN}$), recall ($\frac{TP}{TP+FN}$), precision ($\frac{TP}{TP+FP}$), IoU ($\frac{TP}{TP+FP+FN}$), and F1 score ($\frac{2TP}{2TP+FP+FN}$), measured for full resolution pixels of the ground truth vs. model prediction. T for True and F for False tells if a pixel is classified correctly or incorrectly. P stands for Positive i.e. "Cloud" pixel, while N stands for Negative i.e. "Clear" pixel. So, e.g. True Positive (TP) corresponds to the number of pixel instances that were correctly classified as "Cloud". All metrics are mean values measured over a test set of randomly selected 24 images held out from the training set.

B. Baselines

As a baseline for the proposed method, we consider semantic segmentation algorithms applied on patches, corresponding to the predominant approach to segmentation of large MSI images [18], [19], [10]. To focus on demonstrating the importance of the proposed multi-scale approach, we consider conventional encoder-decoder CNNs as baselines, matching our fine-grained

TABLE III: Comparison of variants of the proposed multi-scale network with different coarse model encoders, both with fine-grained model set to the best CNN baseline of Table II (first three rows; with "Fine" assigned to SEResNeXt-50/FPN as the fine-grained network) and with naively mapping all "Partly Cloudy" patches as "Cloud" (next three rows) to standard cloud detection models tested against our dataset. For reference, we also report the best CNN baseline again.

Model	acc.	prec.	recall	f1	iou
ResNet-50 → Fine	0.810	0.774	0.684	0.680	0.550
InceptionV3 → Fine	0.827	0.800	0.651	0.662	0.542
VGG16 → Fine	**0.856**	0.810	0.711	**0.711**	**0.597**
InceptionV3 → None	0.779	0.634	0.759	0.628	0.510
ResNet-50 → None	0.769	0.632	0.777	0.647	0.516
VGG-16 → None	0.800	0.624	**0.839**	0.666	0.551
FMASK	0.755	0.842	0.446	0.400	0.534
IDEPIX	0.736	**0.944**	0.354	0.351	0.467
SEN2COR	0.688	0.819	0.267	0.255	0.371
CNN baseline	0.737	0.718	0.643	0.617	0.494

component. Since the strength of the baseline depends on the choice of encoder backbone and upsampling architecture, we report results for several alternatives.

The set of available CNN architectures is constantly growing, and the set of their possible combinations would become impossibly large for full enumeration. We include a selection of both state-of-the-art and established CNN architectures that we consider representative and limit the combinations as follows. We initially fix the segmentation architecture to UNet [12], applied to cloud segmentation in recent work [49], while varying the backbone encoders. We consider encoders Inception-v3 [27], ResNet-50 [24], EfficientNet [26], SEResNeXt-50 [25], and VGG-16 [23] in Table I. We then vary the segmentation architecture, evaluating UNet [12], Linknet [22], FPN [21], and PSPNet [20], now fixing the encoder to SEResNeXt-50 that initially performed best with UNet. Table II indicates that in terms of accuracy, best results for the considered encoder-decoder architectures are obtained with a combination of FPN and SEResNext-50. While some variants (UNet/EfficientNet and LinkNet/SEResNext-50) have higher recall, they simultaneously show a significant drop in accuracy.

C. Coarse encoder selection and method validation

For the coarse component to be trained with the undersampled derived dataset of complete images, we considered the same encoder architectures as for the fine-grained model. We modified these for use as the coarse component as described in III-B1 but omitted EfficientNet and SEResNeXt-50 due to technical and memory limitations of the test environment. To demonstrate the effect of prepending our coarse component in a cascade, using otherwise identical choices in the proposed method and a CNN baseline, we assume the FPN/SEResNext-50 as the fine-grained model component, the same network having been evaluated above as a baseline. We additionally report results for a simplified variant that omits the fine-grained model altogether and simply maps all pixels in "Partly Cloudy" patches as "Cloud". This naturally maximizes recall and is
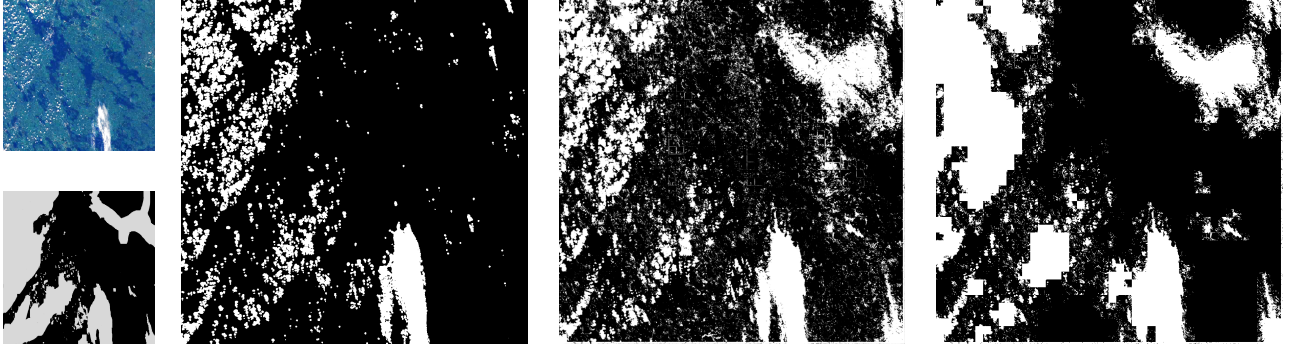
Fig. 6: Segmentation results. The small images to the extreme left show the MSI image and the binary ground truth, followed by the masks produced by the FMASK cloud detection model (left), best CNN baseline (center) and the proposed model (right). The proposed model clearly matches the ground truth most accurately of the three.

included in the results to illustrate the importance of modeling global features.

Table III reports the metrics for the model variants, using the test images. To provide remote sensing context for the results, we report the same metrics also for three well-known MSI cloud masking algorithms (FMask [34], Sen2Cor [50] and Idepix [51])[2]. For the Idepix baseline mask, we include pixels marked as *f_cloud, f_cirrus_sure, f_cirrus_ambiguous, f_clear_snow, f_cloud_shadow or f_brightwhite*, for maximal accuracy and recall against our dataset. For Sen2Cor, we include *"Cloud Shadows", "Clouds (low to high probability)", "Cirrus"* and *"Snow/Ice"*.

Irrespective of the choice of the coarse encoder, the proposed method outperforms all of the existing cloud detection models by a wide margin. We also confirm high recall compared to the alternatives, matching a key motivation of the work. Figure 6 illustrates the difference for a single test image.

For these comparisons, we used the class-wise weighted cross-entropy loss with hyperparameter values of $\alpha = 2$ and $\beta = 0.5$ for the "Partly Cloudy" class and $\alpha = \beta = 1$ for others, setting $\gamma = 0$ for all classes. That is, the best result did not use the adjacency loss, which in our experiments nevertheless can better capture the coarse cloud boundary at a marginal cost on overall accuracy, as shown next.

D. Effect of loss function elements

The motivation to use a specific loss function for training the coarse model, stemming from the inherently small proportion of partly cloudy patches, was analyzed in Section III-C. Table IV measures the effect of the custom loss function elements described, comparing the proposed class-wise weighted cross entropy (WCE) loss with and without the adjacency penalty against the standard choice of categorical cross entropy (CCE) loss. We evaluate the effect of the loss in context of one example configuration, a coarse ResNet-50 with fine-grained ResNet-50/Unet. For WCE we use the same weighting as in Section V-C, and for the adjacency loss weight, we use

TABLE IV: Effect of coarse model loss function choice on internal (coarse-acc) and overall (accuracy) performance, as well as the normalized adjacency score measuring how well the borders between classes are captured. We compare variants of our loss function against standard categorical cross-entropy (CCE).

Loss function	coarse-acc	accuracy	S_{dn}
CCE	0.813	0.718	0.09781
$\mathcal{L}_{wce}$	**0.855**	**0.812**	0.01778
$\mathcal{L}_{adj} + \mathcal{L}_{wce}$	0.846	0.809	**0.00968**

$\gamma = 1$. Since an adjacency loss $\mathcal{L}_{adj}$ term is defined for a pair of classes, we use a separate $\mathcal{L}_{adj}$ term for each class pair out of the three classes, penalizing deviations in the count of "Overcast/Cloudless" pairs with a weight of 0.75 and the other two pairs ("Overcast/Partly Cloudy" and "Cloudless/Partly Cloudy") by 0.125.

The result is that both of the improved loss functions clearly outperform the common CCE in terms of pixel accuracy of the final segmentation (accuracy) as well as the internal accuracy of the coarse model (coarse-acc). Importantly for reducing the unwanted adjacencies of Overcast/Cloudless, the weighted loss functions improve the target metric of normalized adjacency score (S_{dn}) for the class pair roughly by a factor of ten, which indicates that they reproduce the proportions of class boundary regions considerably better. This can be visually verified in Figure 3 that shows sample predictions for each loss. The class-wise weighted cross-entropy in itself already improves boundary detection notably, but interestingly, a further improvement by a factor of two is achieved in discerning the cloud boundary by activating the adjacency term, only negligibly impacting accuracy (0.809 vs. 0.812). This would have made $\mathcal{L}_{adj} + \mathcal{L}_{wce}$ an equally valid basis for the previous experiment on overall performance, providing additional smoothness at mask borders.

E. Computation time

Segmenting clouds on a single high-resolution Sentinel-2 MSI image using the proposed dual model framework, the proposed dual-network architecture ran 4.1 times faster compared to the FPN/SEResNeXt baseline, taking roughly 9s

---

[2]The metrics for these masks are reported at 20m, their highest available common resolution. Note that the impact of undersampling on the numbers is negligible; e.g. the proposed method evaluated at 20m resolution has 0.8567 accuracy compared to the 0.8564 reported for full 10m resolution.

per image vs 37s for the baseline on a NVIDIA GTX10180 GPU. The improvement is explained by a reduction of almost 80% in the number of patches processed. The time overhead added by the coarse model is negligible, less than 1s per image.

F. Discussion

From the point of view of the semantic segmentation problem, the proposed framework outperforms the selected CNN baseline (see Table III) by all chosen metrics for a cloud mask dataset with characteristics of contiguity and enhanced cloud recall. For instance, pixel accuracy was improved from 73.7% to 85.6%. i.e. by a relative accuracy improvement of 16%. In view of the cloud detection application in remote sensing, we achieved a 13% relative improvement in accuracy to the closest considered cloud detection algorithm. This, alongside validating our method of approach, also suggests that the proposed method has potential to narrow down the gap between applications' needs for sensitive cloud masks vs. available cloud masking methods.

Interestingly, our coarse model architecture reaches 80% accuracy also when the fine-grained model is not used at all. In this variant, we set all patches marked as "Partly Cloudy" to contain "Cloud" pixels only and as a result, we outperform the CNN baselines based on patching. This highlights the importance of modeling larger spatial features, especially in tasks where the output needs to be largely contiguous, and suggests steering more effort in MSI segmentation towards models inspecting larger proportions of the whole image in contrast to majority of the current literature; see Section II.

Our task of specifically reproducing the given approximate and contiguous annotations, based both on geospatially widespread and pixel-level information, is more challenging than typical supervised CNN setups where target objects have boundaries, or annotations are focused on the pixel. Still, the overall accuracy can be contrasted to those reported in literature for standard masks. In a recent study, Baetens et al. [5] report accuracies in the range of $84 - 91\%$ for validating a generated reference annotation against FMask, MAJA and Sen2Cor. We reach accuracy that is within the same range in a more challenging task. This validates that the overall architecture and data pipeline reflect the state of the art in the field. On the other hand, we demonstrated that many existing masks are not sufficient for recreating the annotations in our data and in particular have very poor recall, between 0.27 and 0.45, compared to 0.71 (or 0.84 if mapping the whole border area to "Cloud") of the model trained on these annotations.

Absolute accuracy, for developing the framework into a production application, could be further improved e.g. by taking advantage of readily available MSI image QA bands e.g. snow and ice in e.g. preprocessing a subtraction from ground truth mask and treatment as distinct classes, in the style of [38], or by searching for an optimal fine-grained architecture and the hyperparameters α, β, and γ using a more systematic validation procedure.

VI. CONCLUSION

We propose a novel two-phase semantic segmentation framework for cloud detection from high-resolution optical remote sensing images, drawing on state-of-the-art CNN architectures. We train the components of the CNN framework with cloud masks manually annotated with sensitivity for sparse regions of cloud-ambiguous or hazy pixels. Our dataset, originally collected during construction of cloudless mosaics for a land cover project[28], contains images from within the growing season, including bare ground, with residual snow, ice and clouds included in a single class of an exclusion mask. To identify image patches that need the most fine-grained cloud detection, and also to reproduce a contiguous quality present in the given annotated masks, we use a modified VGG architecture at an undersampled input resolution. Here, we use a weighted loss function to handle an inherent class imbalance. To classify pixels of the cloud-ambiguous patches at full resolution, we pass them on to a second CNN component based on a SEResNeXt-FPN architecture [25][21] . The overall framework allows us to analyze large images in a wider variation of scale than otherwise would be possible in the usual setup of a single CNN. Our experiments show a relative accuracy improvement of 16% by our aggregated dual model over baselines of well-known encoder-decoder CNN architectures trained on image patches.

As a semantic segmentation framework of large images, the proposed solution is not limited to the domain of remote sensing or cloud detection. We apply the framework to a high-resolution Sentinel-2 dataset but expect it to be readily applicable to semantic segmentation of other multispectral datasets such as Landsat or MODIS and to perform well especially in scenarios of liberally annotated, highly contiguous masks over features having ambiguous boundaries.

REFERENCES

[1] D. Marmanis, J. D. Wegner, S. Galliani, K. Schindler, M. Datcu, and U. Stilla, "Semantic segmentation of aerial images with an ensemble of cnns," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 3, p. 473, 2016.

[2] M. Längkvist, A. Kiselev, M. Alirezaie, and A. Loutfi, "Classification and segmentation of satellite orthoimagery using convolutional neural networks," *Remote Sensing*, vol. 8, no. 4, p. 329, 2016.

[3] N. Kussul, M. Lavreniuk, S. Skakun, and A. Shelestov, "Deep learning classification of land cover and crop types using remote sensing data," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 5, pp. 778–782, 2017.

[4] S. Mahajan and B. Fataniya, "Cloud detection methodologies: variants and development—a review," *Complex & Intelligent Systems*, pp. 1–11, 2019.

[5] L. Baetens, C. Desjardins, and O. Hagolle, "Validation of copernicus sentinel-2 cloud masks obtained from maja, sen2cor, and fmask processors using reference cloud masks generated with a supervised active learning procedure," *Remote Sensing*, vol. 11, no. 4, p. 433, 2019.

[6] S. Ackerman and R. Frey, "Modis atmosphere l2 cloud mask product (35_l2)," *NASA MODIS Adaptive Processing System, NASA Goddard Space Flight Center: Greenbelt, MD, USA*, 2015.

[7] P. Kamavisdar, S. Saluja, and S. Agrawal, "A survey on image classification approaches and techniques," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 2, no. 1, pp. 1005–1009, 2013.

[8] P. Li, L. Dong, H. Xiao, and M. Xu, "A cloud image detection method based on svm vector machine," *Neurocomputing*, vol. 169, pp. 34–42, 2015.

[9] S. Le Hégarat-Mascle and C. André, "Use of markov random fields for automatic cloud/shadow detection on high resolution optical images," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 64, no. 4, pp. 351–366, 2009.

[10] G. Mateo-García, L. Gómez-Chova, and G. Camps-Valls, "Convolutional neural network for multispectral image cloud masking," in *2017 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2017, pp. 2255–2258.

[11] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[12] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[13] J. Drönner, N. Korfhage, S. Egli, M. Mühling, B. Thies, J. Bendix, B. Freisleben, and B. Seeger, "Fast cloud segmentation using convolutional neural networks," *Remote Sensing*, vol. 10, no. 11, p. 1782, 2018.

[14] M. Shi, F. Xie, Y. Zi, and J. Yin, "Cloud detection of remote sensing images by deep learning," in *2016 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2016, pp. 701–704.

[15] F. Xie, M. Shi, Z. Shi, J. Yin, and D. Zhao, "Multilevel cloud detection in remote sensing images based on deep learning," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 10, no. 8, pp. 3631–3640, 2017.

[16] T. Chen, B. Xu, C. Zhang, and C. Guestrin, "Training deep nets with sublinear memory cost," *arXiv preprint arXiv:1604.06174*, 2016.

[17] L. Wang, J. Ye, Y. Zhao, W. Wu, A. Li, S. L. Song, Z. Xu, and T. Kraska, "Superneurons: dynamic gpu memory management for training deep neural networks," in *Proc. of the 23rd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 2018, pp. 41–53.

[18] Z. Shao, Y. Pan, C. Diao, and J. Cai, "Cloud detection in remote sensing images based on multiscale features-convolutional neural network," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 6, pp. 4062–4076, 2019.

[19] J. Yang, J. Guo, H. Yue, Z. Liu, H. Hu, and K. Li, "Cdnet: Cnn-based cloud detection for remote sensing imagery," *IEEE Transactions on Geoscience and Remote Sensing*, 2019.

[20] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2881–2890.

[21] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[22] A. Chaurasia and E. Culurciello, "Linknet: Exploiting encoder representations for efficient semantic segmentation," in *Proc. of IEEE Visual Communications and Image Processing*. IEEE, 2017, pp. 1–4.

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[25] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[26] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *arXiv preprint arXiv:1905.11946*, 2019.

[27] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[28] "Corine land cover," https://land.copernicus.eu/user-corner/publications/clc-flyer/at_download/file, accessed: 2019-11-18.

[29] "Copernicus cloud masking assessment," https://land.copernicus.eu/news/cloud-masking-assessment, accessed: 2019-11-18.

[30] S. A. Ackerman, K. I. Strabala, W. P. Menzel, R. A. Frey, C. C. Moeller, and L. E. Gumley, "Discriminating clear sky from clouds with modis," *Journal of Geophysical Research: Atmospheres*, vol. 103, no. D24, pp. 32 141–32 157, 1998.

[31] L. L. Stowe, C. Wellemeyer, H. Yeh, and T. Eck, "Nimbus-7 global cloud climatology. part i: Algorithms and validation," *Journal of Climate*, vol. 1, no. 5, pp. 445–470, 1988.

[32] J. Cihlar and J. Howarth, "Detection and removal of cloud contamination from avhrr images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 32, no. 3, pp. 583–589, 1994.

[33] R. R. Irish, J. L. Barker, S. N. Goward, and T. Arvidson, "Characterization of the landsat-7 etm+ automated cloud-cover assessment (acca) algorithm," *Photogrammetric engineering & remote sensing*, vol. 72, no. 10, pp. 1179–1188, 2006.

[34] Z. Zhu, S. Wang, and C. E. Woodcock, "Improvement and expansion of the fmask algorithm: Cloud, cloud shadow, and snow detection," *Remote Sensing of Environment*, vol. 159, pp. 269–277, 2015.

[35] S. Metsämäki, J. Pulliainen, M. Salminen, K. Luojus, A. Wiesmann, R. Solberg, K. Böttcher, M. Hiltunen, and E. Ripper, "Introduction to globsnow snow extent products with considerations for accuracy assessment," *Remote Sensing of Environment*, vol. 156, pp. 96–108, 2015.

[36] S. Foga, P. L. Scaramuzza, S. Guo, Z. Zhu, R. D. Dilley Jr, T. Beckmann, G. L. Schmidt, J. L. Dwyer, M. J. Hughes, and B. Laue, "Cloud detection algorithm comparison and validation for operational landsat data products," *Remote sensing of environment*, vol. 194, pp. 379–390, 2017.

[37] A. Visa, K. Valkealahti, and O. Simula, "Cloud detection based on texture segmentation by neural network methods," in *Proc. of IEEE International Joint Conference on Neural Networks*. IEEE, 1991, pp. 1001–1006.

[38] S. Mohajerani, T. A. Krammer, and P. Saeedi, "Cloud detection algorithm for remote sensing images using fully convolutional neural networks," *arXiv preprint arXiv:1810.05782*, 2018.

[39] M. Segal-Rozenhaimer, A. Li, K. Das, and V. Chirayath, "Cloud detection algorithm for multi-modal satellite imagery using convolutional neural-networks (cnn)," *Remote Sensing of Environment*, vol. 237, p. 111446, 2020.

[40] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[41] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[42] H. Liu, D. Zeng, and Q. Tian, "Super-pixel cloud detection using hierarchical fusion cnn," in *Proc. of IEEE International Conference on Multimedia Big Data*. IEEE, 2018, pp. 1–6.

[43] H. Miyamoto, K. Uehara, M. Murakawa, H. Sakanashi, H. Nasato, T. Kouyama, and R. Nakamura, "Object detection in satellite imagery using 2-step convolutional neural networks," in *2018-2018 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2018, pp. 1268–1271.

[44] G. Papandreou, L.-C. Chen, K. P. Murphy, and A. L. Yuille, "Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation," in *Proc. of the IEEE international conference on computer vision*, 2015, pp. 1742–1750.

[45] O. Z. Kraus, J. L. Ba, and B. J. Frey, "Classifying and segmenting microscopy images with deep multiple instance learning," *Bioinformatics*, vol. 32, no. 12, pp. i52–i59, 2016.

[46] R. Shen, T. Guthier, H. Mobis, B. Tang, and I. B. Ayed, "Scribble supervised annotation algorithms of panoptic segmentation for autonomous driving."

[47] D. Pathak, E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional multi-class multiple instance learning," *arXiv preprint arXiv:1412.7144*, 2014.

[48] P.-A. Ganaye, M. Sdika, and H. Benoit-Cattin, "Semi-supervised learning for segmentation under semantic constraint," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2018, pp. 595–602.

[49] J. H. Jeppesen, R. H. Jacobsen, F. Inceoglu, and T. S. Toftegaard, "A cloud detection algorithm for satellite imagery based on deep learning," *Remote sensing of environment*, vol. 229, pp. 247–259, 2019.

[50] R. Richter, J. Louis, and U. Müller-Wilm, "Sentinel-2 msi—level 2a products algorithm theoretical basis document," *European Space Agency,(Special Publication) ESA SP*, vol. 49, no. 0, pp. 1–72, 2012.

[51] C. Lebreton, K. Stelzer, C. Brockmann, L. Bertels, N. Pringle, M. Paperin, O. Danne, E. Knaeps, and K. Ruddick, "Cloud and cloud shadow masking of high and medium resolution optical sensors," in *Living Planet Symposium*, vol. 740, 2016, p. 407.

Supplementary material for:
Multi-scale cloud detection in remote sensing images using a dual convolutional neural network

Markku Luotamo*, Sari Metsämäki and Arto Klami

I. INTRODUCTION

This supplementary material provides additional technical details for a manuscript with the same title. Section II explains the full data processing pipeline and provides justification for the various resolution choices presented in the manuscript. Section III presents technical details for the differentiable approximation used for constructing the loss function for the model. Finally, Section IV presents a full specification for one of the neural network architecture baselines used for pixel-level segmentation as the "fine-grained model".

II. DATA PIPELINE DIMENSIONS

The manuscript as such provides a high-level overview of the data pipeline, for brevity skipping rationale and details. In this section, we cover the remaining details and calculate the dimensions in a reverse order from end to beginning, from a computationally convenient patch size of the fine-grained model (see Fig. 2) towards the constraint of a Sentinel-2 MSI resolution of 10980x10980. This yields the dimensions of the coarse model used earlier in the processing steps (See Fig. 1). The details are presented for the specific input size, but we note that a similar procedure could easily be carried out for images of other dimensionalities.

For training deep neural networks on a GPU, input image size of a multiple of 8 is commonly used and recommended in the industry (see for instance [1]). We apply this recommendation with Tensorflow and an NVidia GPU GTX1080 in our use: we select 256x256, an increase from the most common 224x224 patch size to allow slightly larger features from the start. We reserve a 4-pixel margin from the patch for overlap to overcome edge artifacts we observed in early experiments. This results in an effective patch size of 248x248. Dimensions of Sentinel-2 MSI data, 10980x10980 px, are not divisible by 248 (or 256 or 224, for that matter), so the images and the ground truth need to be appended with slices we call a completion pad, to 11160x11160 px in order to be the closest multiple of 248 (45*248=11160). We can now slice the 11160x11160 data to 45x45 patches of 248x248 px effective size, that is, 256x256px with the overlap pixels, ready to be input to the fine-grained model. In the end we hence use 256x256x7 slices of the original image and 256x256x3 masks as data input.

Let us then assume a 45x45 bottleneck resolution for our coarse model. 32:1 is the spatial ratio found in many CNN segmentation encoder backbones between input resolution and the bottleneck layer for height and width. When 45x45 is set as the bottleneck resolution, we get an input resolution of 1440x1440px for the coarse model. On the other hand 1440x1440 approaches the empirical upper limit for 7-band image size for the given GPU memory size and suggested neural network. We downsample the 11160x11160 image to 1440x1440, which provides the coarse model with input.

The coarse model ground truth masks, however, need to be of shape 45x45x4 for classifying each patch of 248x248 on the completion-padded image. These are obtained from the completion-padded 11160x11160x3 ground truth by slicing it to a grid of 45x45x248x248x3 (no overlap). The 248x248 dimensions are eliminated and the last dimension of 3 classes transformed to 4 by pooling the "cloudy" pixels of a patch to four classes (fully overcast/partly cloudy/cloudless/no data) based on pixel count per original class annotation within a patch. The full-resolution "cloudy" pixel count n_{pc} condition for a ground truth patch to be automatically annotated to "partly cloudy" is $0 < n_{pc} < w \times h$ for our experiments below, where w and h are width and height of the patch, i.e. 248x248.

Altogether, this data pipeline supports conditional two-phase processing based on the result of the coarse classification as explained in the article. Only "partly cloudy" patches on the edges of cloudy regions need to be delegated for more granular classification of the patch, to the fine-grained model.

III. DIFFERENTIABLE ONE-HOT ENCODING

Neural networks are trained with gradient-based methods, which implies that the loss function must be differentiable. Besides approximating the step function as explained in the manuscript, we replace the one-hot encoding $\sigma()$ with a soft approximation presented as TensorFlow code in Figure 3.

IV. FINE-GRAINED MODEL ARCHITECTURE

In the article, our work evaluated 11 combinations of different CNN architectures, for brevity omitting experiments with less relevant results. As a representative and straightforward example to illustrate the relationship and propagation of the dimensions from the input images to e.g. the bottleneck layer, we show in Figure 4 a UNet [2], with a ResNet [3] backbone with all of the layer sizes embedded into the figure. The networks are also representative due to their archetypal use

*M.Luotamo and A.Klami: University of Helsinki, Dept. of Computer Science
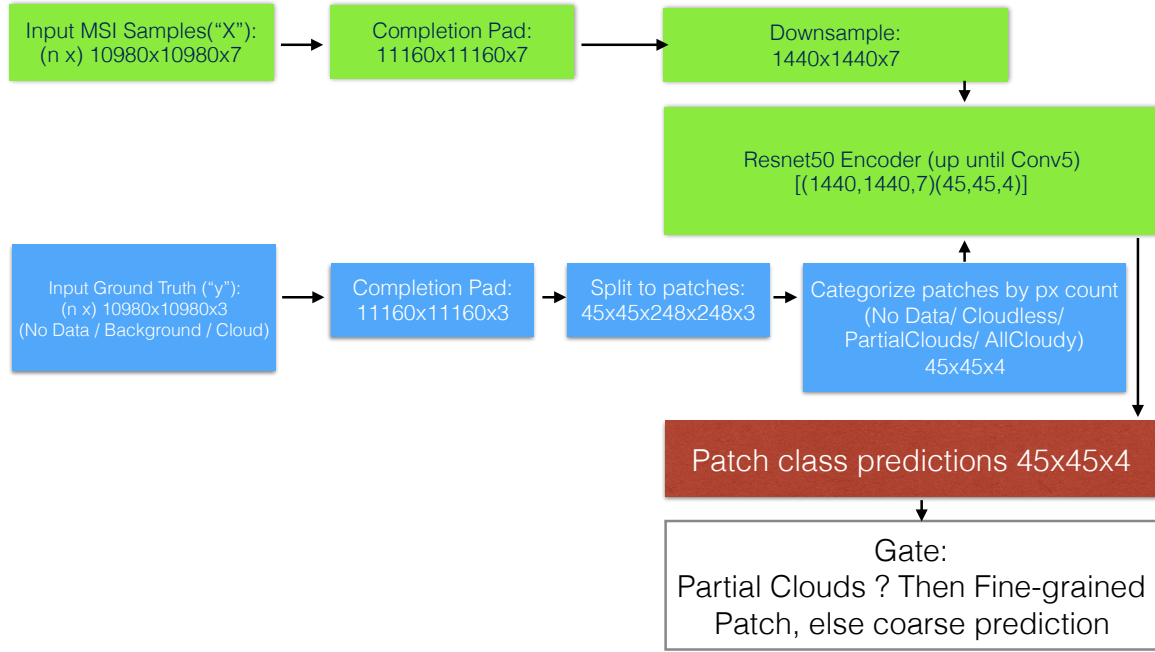Sari Metsämäki: Finnish Environment Institute

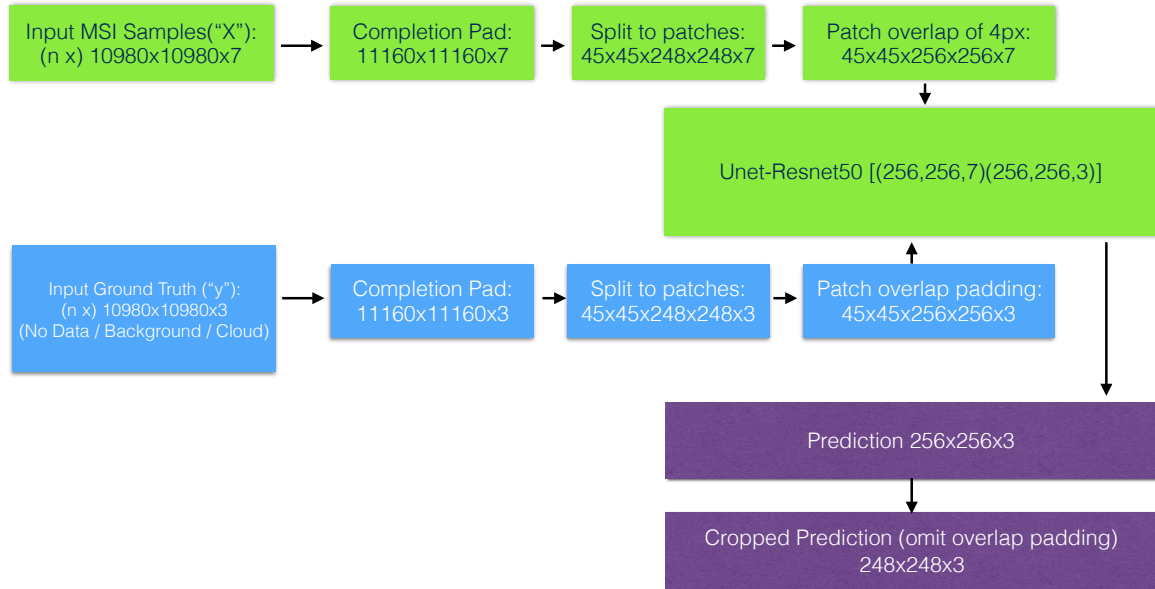Fig. 1: Data pipeline for the coarse model component.

Fig. 2: Data pipeline for the fine-grained model component.

```python
def soft_onehot(x):
    return -(tf.sign(tf.reduce_max(x, axis=-1, keepdims=True) - x) - 1)
```

Fig. 3: TensorFlow code for a differentiable one-hot function used in the coarse model loss function.
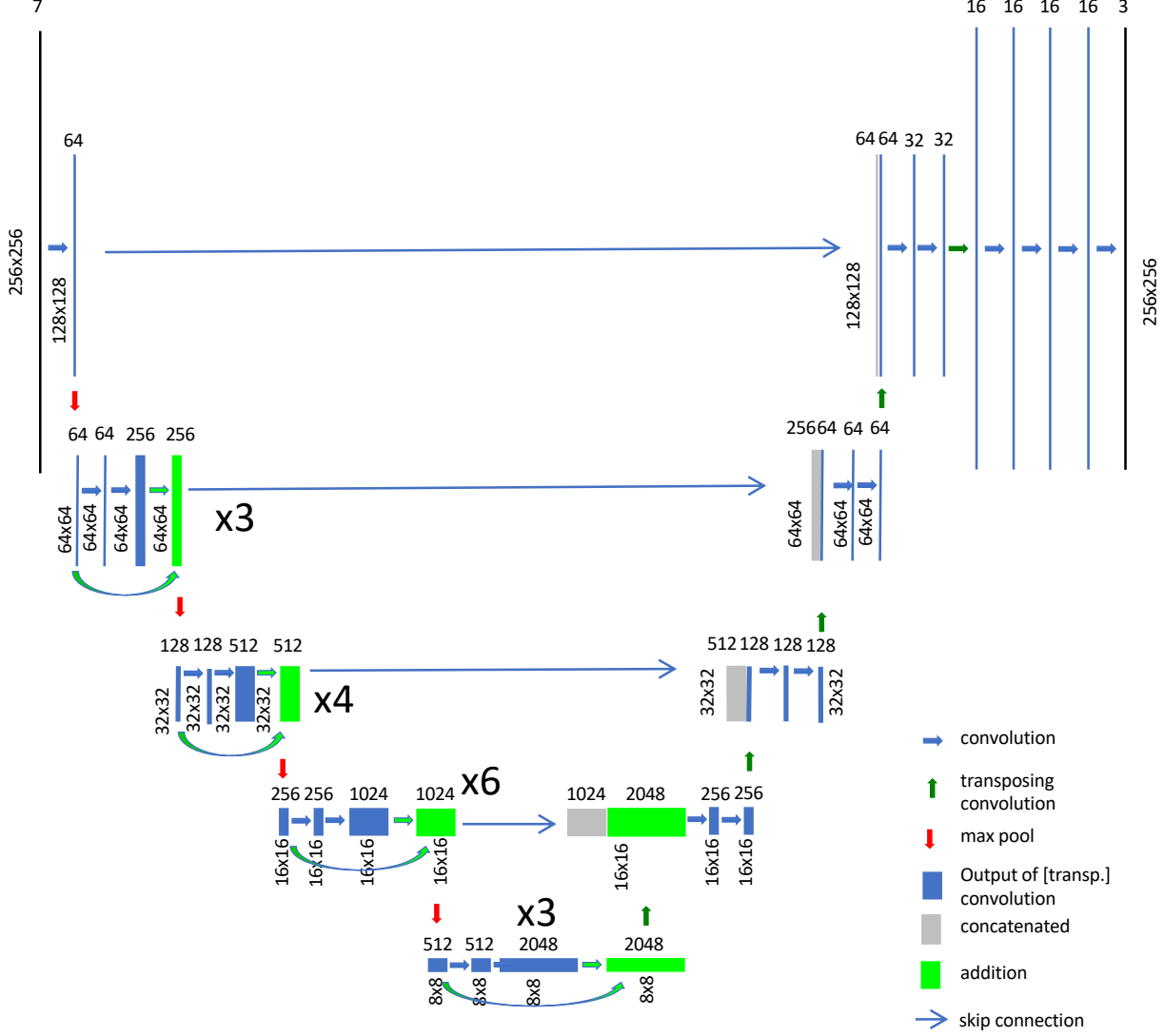

Fig. 4: A possible fine-grained segmentation model structure that draws on U-Net, with a Resnet-50 encoder backbone

of lateral and residual skip connections often present in the evaluated networks from a later date. To make alternative component architectures readily interchangeable, most base implementations were used from a Segmentation models library by Yakubovskiy [4]. The modifications required by the coarse component included first truncating a backbone network from the end up to the encoder bottleneck layer, then appending 1x1 convolutions and a softmax layer to yield a 45x45 classification as explained with more context in the article.

REFERENCES

[1] NVidia, "Deep learning performance guide," https://docs.nvidia.com/deeplearning/sdk/dl-performance-guide/index.html, accessed: 2019-01-13.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[4] P. Yakubovskiy, "Segmentation models (python deep learning library)," https://github.com/qubvel/segmentation_models, accessed: 2019-01-13.