Effective Semantic Pixel labelling with Convolutional Networks and Conditional Random Fields

Sakrapee Paisitkriangkrai*[1], Jamie Sherrah†[2], Pranam Janney‡[2] and Anton Van-Den Hengel§[1]

[1]Australian Centre for Visual Technology (ACVT), The University of Adelaide, Australia
[2]Defence Science and Technology Organisation (DSTO), Australia

Abstract

Large amounts of available training data and increasing computing power have led to the recent success of deep convolutional neural networks (CNN) on a large number of applications. In this paper, we propose an effective semantic pixel labelling using CNN features, hand-crafted features and Conditional Random Fields (CRFs). Both CNN and hand-crafted features are applied to dense image patches to produce per-pixel class probabilities. The CRF infers a labelling that smooths regions while respecting the edges present in the imagery. The method is applied to the ISPRS 2D semantic labelling challenge dataset with competitive classification accuracy.

1. Introduction

Automated annotation of urban areas from overhead imagery plays an essential role in many photogrammetry and remote sensing applications, *e.g.*, environmental modelling and monitoring, building and updating a geographical database, gathering of military intelligence, infrastructure planning, land cover and change detection. Pixel labelling of aerial photography is one of the most challenging and important problems in remote sensing. The objective of pixel labelling is to assign an object class to each pixel in the given image. The task is challenging due to the heterogeneous appearance and high intra-class variance of objects such as building, streets, trees and cars. Although many different algorithms have been proposed in the past [18, 21, 22], the pixel labelling task cannot be considered a solved problem. In this paper, we present a framework to perform semantic pixel labelling and discuss its performance on the ISPRS 2D semantic labelling challenge data set [1].

Semantic labelling is typically applied to multimedia images, and involves dense classification followed by smoothing, for example with a probabilistic graphical model. The traditional visual bag-of-words approach [25] extracts hand-crafted features which are clustered to form visual words, and boosting is used for classification. The success of this method relies on the initial choice of features. More recently, deep CNNs have been used to learn discriminative image features that are more effective than hand-crafted ones. CNNs have been used for semantic labelling of street scenes in [9].

In this paper we apply CNNs to the ISPRS 2D semantic labelling contest for aerial imagery. We choose CNN due to the following reasons. First CNN features can be extracted efficiently. We design our framework such that the entire test image is forward propagated only once. Since overhead images are typically very large, computational efficiency is a priority. Second, by augmenting the training data with various transformations, the CNN representation can be robust to both translation and rotation. Since imagery could be captured at arbitrary azimuth, the feature representation needs to be rotation-invariant. Complementary to learned visual features, we make use of simple hand-crafted features proposed in a previous submission to the labelling challenge [11]. Combining both CNN features with simple hand-crafted features further boosts the labelling accuracy of our proposed approach.

As a post-processing step, a CRF is applied to the label probabilities. The CRF infers a globally-consistent labelling that is locally smooth except at edges in the imagery and can improve fragmented and marginal regions. In previous work on the ISPRS challenge [11], super-pixel CRFs were found not to increase accuracy. In our work we use a pixel-level CRF to avoid errors in over-segmentation, and find that both the accuracy and visual appeal of the labelling

*sakrapee.paisitkriangkrai@adelaide.edu.au
†jamie.sherrah@dsto.defence.gov.au
‡pranam.janney@dsto.defence.gov.au
§anton.vandenHengel@adelaide.edu.au

improve somewhat.

In this paper, we explore the possibilities of using a combined approach for semantic labelling of overhead imagery: learned features complemented by hand-crafted features. Subsequent sections detail the two approaches, experiments and our observations. We also demonstrate the utility of combining both approaches.

2. Related work

Several researchers have applied machine learning in order to annotate overhead imagery. During early days, discrete class labels at a pixel were predicted using a vector of features at each pixel[6, 2, 16, 3]. However, due to lack of high-resolution imagery most of these techniques were predominantly used for terrain classification, *i.e*. classifying an overhead image into forest, water, agricultural land *etc*. Over recent years, advances in digital photography have made available large amounts of high resolution aerial imagery. With high resolution data there is now an opportunity for fine-grained classification such as roads, buildings, and objects such as cars and ships using sophisticated features and machine learning algorithms[20, 21, 22].

Of late in computer vision, CNN features have been shown to outperform traditional hand-crafted features in visual recognition challenges [14], image classification [23] and object detection [12]. CNNs roughly mimic the nature of the mammalian visual cortex and are among the most promising architectures for vision applications. The CNN architecture exploits the strong spatially local correlation present in natural images by enforcing a local connectivity pattern between neurons of adjacent layers. A deep CNN consists of multiple layers of small neuron collections which offer an alternative approach to learn visual patterns directly from raw image pixels.

Recently Razavian *et al*. [23] have shown that training a linear SVM classifier on CNN feature representation achieves superior results compared to highly tuned state-of-the-art systems in all visual classification tasks on various data sets [23]. Logistic regression is a widely used technique in statistics and has received a lot of attention due to its close relations to other large margin classifiers, *e.g*., Support Vector Machine [26] and AdaBoost [10]. These classifiers are well-studied and have been shown to achieve good generalization capability in practice.

The combination of CNNs and CRFs has previously been applied to semantic labelling. In [9] a multi-scale CNN is applied to street scenes for dense classification, and labelling using a CRF defined over super-pixels. Mnih and Hinton learn discriminative image features using deep neural networks to detect roads and buildings from noisy labels [19, 18]. Labels were post-processed using a CRF defined over pixels.

Semantic labelling has been applied to aerial imagery previously. [13] used super-pixel features and CRFs for building detection in aerial imagery. A previous benchmark for the ISPRS labelling challenge [11] used hand-crafted texture and multi-spectral features for classification by AdaBoost. Labelling was performed on super-pixels. They found that post-processing with a CRF decreased accuracy but de-speckled the output.

3. Approach

In this section, we introduce the framework for automated pixel classification in high-resolution aerial images. We first introduce the neural networks adopted for dense feature extraction. We then discuss how we complement CNN features with hand-crafted features to further improve the classification accuracy. Finally we briefly introduce the concept of Conditional Random Fields (CRF) to smooth the final pixel labeling results. An overview of the proposed semantic pixel labeling framework is illustrated in Fig. 1.

3.1. Pixel classification with convolutional network

The convolutional feature classifier is applied densely over the input image. The classifier has two components: a CNN consisting only of convolutional layers, and a logistic regression classifier that takes convolutional features as input and outputs class probabilities. The convolutional features are learned by supervised training of a CNN classifier (described next), and discarding the fully-connected layers of the network leaving only the convolutional layers.

CNN Training for Feature Learning In this paper, we train a CNN by adopting the approach of [14], *i.e*., the CNN network consists of several convolutional layers, which are placed alternatively between contrast normalization and max-pooling layers. Each convolutional layer computes the convolutions between the input and a set of filters. The activation function (rectified linear unit - ReLU) perform a non-linear transformation while the max-pooling layer subsamples the output of the convolutional layer. These two operations improve the robustness of the network to distortions and small translations [14]. The output of the last fully-connected layer is fed to a k-way soft-max layer which produces a distribution over k class labels. All network parameters are learned in a supervised manner.

Dense neural pattern training We follow the work of Razavian *et al*. by applying a multi-class classifier on the CNN feature representation [23]. We consider the multi-class classification problem as a set of binary classification problems. We adopt a simple logistic regression with a 'one-versus-all' scheme as it has been shown to be as accurate as any other multi-class algorithms despite its simplicity [24]. The logistic regression solves the following

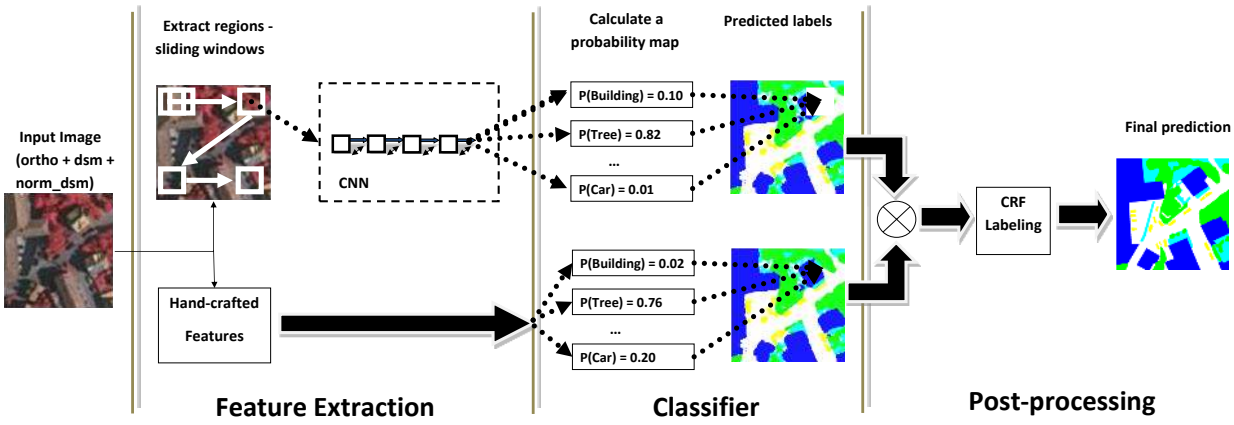


Figure 1: An overview of the proposed pixels labeling framework.

optimization problem,

$$\min_{\boldsymbol{w}_r} \sum_{i=1}^{m} \log\big(1+\exp(-y_i \boldsymbol{w}_r^\top \boldsymbol{x}_i)\big), \tag{1}$$

where $\boldsymbol{w}_r$ is the weight vector we are trying to learn. Here we assume that $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$ is the set of training data, $\boldsymbol{x}_i \in \mathbb{R}^d$ represents vectorized CNN features, $y_i = 1$ if the class label of $\boldsymbol{x}_i$ is the same as r and $y_i = -1$, otherwise. m is the number of training samples, d is the dimension of feature vector $\boldsymbol{x}_i$ and $r \in \{1, 2, \cdots, 5\}$, which corresponds to the class labels *Impervious surfaces*, *Building*, *Low vegetation*, *Tree* and *Car* respectively. In order to avoid overfitting, we introduce ℓ_2-norm regularization on the weight vector $\boldsymbol{w}_r$. Given a test sample $\boldsymbol{x}_t$ and the learned weight vectors $\{\boldsymbol{w}_1, \cdots, \boldsymbol{w}_5\}$, the probability that $\boldsymbol{x}_t$ belongs to class r is given by,

$$P(y_t = r) = \frac{1}{Z}\left(\frac{1}{1+\exp(-\boldsymbol{w}_r^\top \boldsymbol{x}_t)}\right), \tag{2}$$

where $Z = \sum_{r=1}^{5} \frac{1}{1+\exp(-\boldsymbol{w}_r^\top \boldsymbol{x}_t)}$. The purpose of Z is to ensure that the resulting distribution is a probability distribution.

Multi-resolution CNN In order to correctly classify both coarse-scale and fine-scale details in the image, we train several CNN models with different input image resolutions. Each CNN model encodes different patches of increasing sizes, covering a larger context surrounding the centre pixel. The output is a series of feature vectors generated from patches of multiple sizes centred at each pixel (see Fig. 2). A similar concept has also been applied in [8, 9], in which the authors demonstrate that a multi-scale `ConvNet` outperforms a single-scale `ConvNet` for scene parsing.

Implementation We train the CNN with a combination of input data: orthophoto, Digital Surface Model (DSM) image and normalized DSM image. We train three CNN models with three different input image resolutions: 16×16, 32×32 and 64×64 pixels. The parameter settings used in our CNN model for 64×64 pixel input images are explained below. The first convolutional layer filters the $64 \times 64 \times 5$ input image which consists of orthophotos, DSM images and normalized DSM images with 32 kernels of size $5 \times 5 \times 5$ with a stride of 1 pixel. The second convolutional layer takes as input the output of the first convolutional layer and filters it with 64 kernels of size $5 \times 5 \times 32$. The third convolutional layer has 96 kernels of size $5 \times 5 \times 64$ connected to the output of the second convolutional layer. The fourth convolutional layer has 128 kernels of size $3 \times 3 \times 96$. The fully connected (fc) layers have 128 neurons each. We apply the dropout term in both fully connected layers. The dropout term set the output of each hidden neuron to zero with probability of 0.5. We train CNN with stochastic gradient descent at a learning rate of 0.001. The learning rate is reduced by a factor of ten at every 20 epochs. We set the momentum to 0.9 and the weight decay parameters to

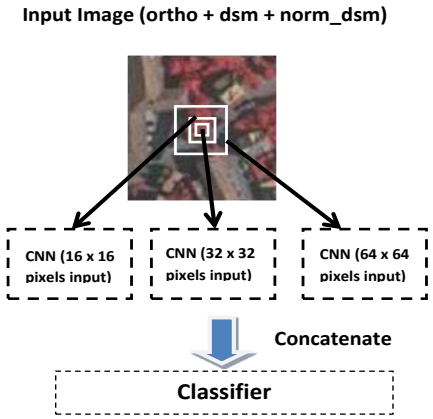


Figure 2: An overview of the multi-resolution CNN.

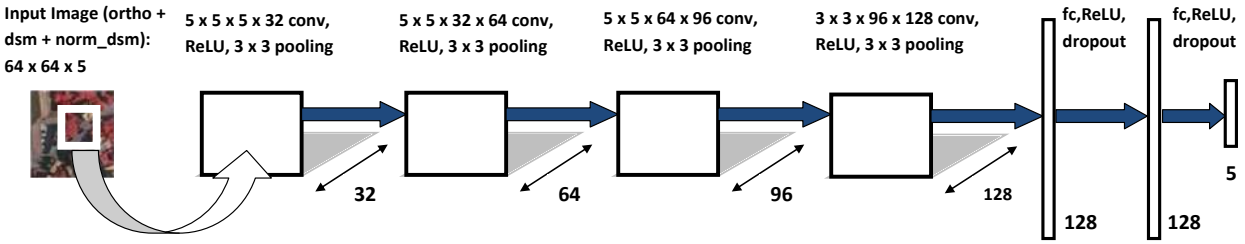


Figure 3: An illustration of the CNN architecture. In this figure, the networks input is the $64 \times 64 \times 3$-pixels orthophoto, 64×64-pixels DSM input image and 64×64-pixels normalized DSM image. The networks consist of six layers (four convolutional layers and two fully-connected layers) with a final 5-way soft-max layer.

0.0005. An illustration of the CNN structure is shown in Fig. 3.

To train the CNN model for 16×16 and 32×32 pixels input images, we simply replace the input image to the first convolutional layer from $64 \times 64 \times 5$ to $16 \times 16 \times 5$ and $32 \times 32 \times 5$, respectively. All other parameter settings are kept the same as before. In this paper, we implement the network training based on the MatConvNet CNN toolbox[1]. Training is done on a standard desktop with an NVIDIA GTX 780 GPU with 6 GB memory. We randomly extract 7500 patches from each class. The time to train the CNN model is under two hours. For ℓ_2-regularized logistic regression we use LIBLINEAR [7].

Dense neural pattern classification The output of each convolution kernel is a dense feature map of neural patterns over the entire image. By carefully designing the CNN structure, we can compute the location of each neural pattern from a patch and map it back to coordinates on the original image [27]. Given the test image and the trained CNN model, we extract the dense CNN feature map from the output of the last convolutional layer. We vectorize CNN features within each input image patch, concatenate them into a single feature vector and apply logistic regression weights to classify object of different classes. To evaluate the entire test image, we adopt a scanning window approach with a step size of 4 pixels. Since we forward propagate the entire test image through the CNN structure only once, we significantly speed up the CNN feature extraction time during evaluation. In addition, since the only class-specific computation is the dot product between extracted CNN features and logistic regression weights, our approach can scale to hundreds of object classes.

3.2. Classification using Hand-Crafted Features

In [11], several pixel-level features were found to be effective for discriminating the classes in the labeling contest. Since these are complementary to the texture-based features extracted by the CNN, a separate random forest (RF) classifier is trained on hand-crafted features and the output probabilities combined with those generated by the CNN. For each pixel in the imagery, a vector of 7 features is generated: NDVI, saturation and normalised DSM (see [11]), $\frac{\text{NIR+R+G}}{3}$, the 3-channel maximum to indicate shadows, entropy and kurtosis of l_2 normalised *histogram of normals* gathered over a 16×16 neighbourhood from the DSM and normalised DSM. *Histogram of normals* is a collation of angles of point normals into 2D histogram bins *i.e.* elevation and azimuth.

537000 training examples are chosen at random and used along with the corresponding ground truth pixel labels to train a random forest classifier with 100 trees. The *Car* class is excluded from this classifier since the features are not discriminative for cars.

Since the CNN and RF are such different approaches, we assume they are independent given the data and multiply their class probabilities to result in the combined probability for each class:

$$p_i^{\text{combo}} = \frac{p_i^{\text{cnn}}\, p_i^{\text{rf}}}{\sum_{j=1}^{C} p_j^{\text{cnn}}\, p_j^{\text{rf}}} \tag{3}$$

where p^{combo}, p^{cnn} and p^{rf} are the combined, CNN and random forest probabilities per class. For the *Car* class the combined probabilities come from the CNN only.

3.3. CRF Labelling

A *conditional random field* (CRF) is a probabilistic graphical model that has been used extensively for semantic labelling of images, for examples see [25, 9]. CRFs are often defined at the super-pixel level rather than the pixel level to improve computational efficiency and robustness [13]. As pointed out in [11], this places an upper limit on the achievable accuracy due to over-segmentation errors (i.e. super-pixels that cover multiple objects). Therefore we use a pixel-level CRF: a 4-connected grid in which each node corresponds to the class label of an image pixel.

Following the standard definition of image labelling CRFs, the energy function consists of unary and pairwise

[1] http://www.vlfeat.org/matconvnet

cost terms:

$$E = \sum_{i \in \mathcal{V}} \Phi(c_i, \boldsymbol{x}) + \sum_{i,j \in \mathcal{E}} \Psi(c_i, c_j, \boldsymbol{x}) \quad (4)$$

where $\mathcal{V}$ and $\mathcal{E}$ are the nodes and edges of the CRF graph, c_i is the class label of node i and $\boldsymbol{x}$ represents the given data. The unary cost is based on the class probability from the combined CNN and RF classifiers:

$$\Phi(c_i, \boldsymbol{x}) = -\log p_{c_i}^{combo} \quad (5)$$

The pairwise costs use a contrast-sensitive Potts model to penalise class boundaries with low contrast. However the traditional method comparing the neighbouring pixel intensities is problematic due to low contrast edges in the image. Instead, pairwise costs are based on a binary edge image such that class boundaries are encouraged to line up with the edges. This provides the CRF with stronger information about class boundaries than the weak image contrast information. Suppose we have a binary edge image $G(i)$ that is true if pixel i is an edge pixel and false otherwise. Define $B(i,j)$ as an indicator that pixel i and its 4-connected neighbour j straddle an edge:

$$B(i,j) = \begin{cases} 1 & \text{if } G(i) \text{ and } \neg G(j) \text{ and} \\ & (x(j) > x(i) \text{ or } y(j) > y(i)) \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

where $x(i)$ and $y(i)$ are the image column and row of pixel i. This asymmetric definition stops edge pixels from being segmented as regions.

The pairwise cost is defined as:

$$\Psi(c_i, c_j, \boldsymbol{x}) = \begin{cases} K(1 - B(i,j)) & \text{if } c(i) \neq c(j) \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

where K is a constant penalty term. In our eperiments the value of $K = 25$ is chosen to minimise the validation set error. The Canny edge detector is used to form the robust edge image $G(i)$; hysteresis thresholding fills in weak edges connecting strong ones, and discards isolated weak edges. Since the Canny image edges are sometimes absent or incomplete, a combined edge map is used. Canny edges are computed separately for the (greyscale) orthophoto, DSM and NDVI, and the three image sets are superimposed to form $G(i)$. Figure 4 shows the complementary nature of the three sets of edges: image edges are accurate but unreliable, DSM edges are more reliable but less accurate, and NDVI edges can delineate vegetation well regardless of elevation.

An approximation to the MAP labelling is inferred using $\alpha - \beta$ swaps [5]. This method solves the n-labelling problem by iteratively solving for the simpler binary case. The maxflow algorithm of [4] is used for binary labelling, which performs exact inference in polynomial time. The CRF was implemented in python and C++[2].

[2] http://github.com/RockStarCoders/alienMarkovNetworks

4. Experiments

The proposed method was applied to the ISPRS labelling contest dataset[1]. The dataset consists of 33 large image patches of different sizes, each being a true orthophoto(TOP) extracted from a larger TOP image captured over Vaihingen, Germany. In total there are over 168 million pixels. The dataset also contains corresponding Digital Surface Models(DSM) for each patch. The patches have a ground sampling distance of 9 cm and the DSMs were generated via dense image matching. Labelled ground truth was provided for 16 of the areas, and were made up of 6 categories: *Impervious surfaces*, *Building*, *Low vegetation*, *Tree*, *Car* and *Clutter/background*. Normalised DSMs was provided to us at a later date, and were generated using the lasground tool[3] where the normalized height is computed based on the off-ground pixels. The effect of terrain or ground is nullified in normalised DSM compared to the regular DSM. Guidelines for evaluation procedure and metrics are defined by the ISPRS[1].

We investigate the experimental design of our approach. We split the labelled training images into training and validation sets. The training set consists of 11 areas (1, 3, 5, 7, 13, 17, 21, 23, 26, 32, 37) and the validation set consists of 5 areas (11, 15, 28, 30 and 40). The evaluation is based on the computation of pixel-based confusion matrices. For each class, we report the harmonic mean of precision and recall (F1-score). We also report the overall accuracy (Overall Acc.), which is the normalized trace from the confusion matrix (*i.e.* percentage of pixels correctly labelled). As per the ISPRS metrics, pixels near ground truth class boundaries are excluded by eroding the labels with a 5×5 diamond.

Input data In this experiment, we compare the CNN performance with and without the DSM model. All experimental settings are kept identical, except the number of channels of convolutional kernels in the first layer. For orthophotos, the filter size in the first layer is set to $5 \times 5 \times 3 \times 32$. For orthophotos + DSM, the filter size in the first layer is set to $5 \times 5 \times 4 \times 32$. We conduct experiments with both raw DSM and the normalized DSM. Table 1 compares the average F1-score and overall accuracy of the CNN given different input data. We observe that it is beneficial to use the normalize height information as it improves the overall accuracy by 3.3%. Not surprisingly, we observe that the normalized height feature has less impact on the detection rate of car-pixels (53.5% versus 54.6%). A similar finding has also been reported in [11]. In our experiment, we achieve the highest accuracy when we combine orthophoto with the raw DSM and the normalized DSM (an improvement of 5.2% on overall accuracy).

[3] http://rapidlasso.com

Figure 4: Example of edges used in CRF pairwise cost.

	Imp. surf.	Building	Low veg.	Tree	Car	Average F1	Overall Acc.
Ortho	82.91%	88.13%	67.14%	81.77%	53.50%	74.69%	80.10%
Ortho + DSM	82.93%	88.75%	68.40%	82.44%	51.15%	74.74%	80.71%
Ortho + NDSM	86.07%	92.79%	72.85%	82.85%	54.63%	77.84%	83.46%
Ortho + DSM + NDSM	87.35%	93.34%	74.96%	84.97%	63.32%	80.79%	**85.18%**

Table 1: Performance comparison of the CNN with different input data sources (Ortho - Orthophoto, DSM - Raw Digital Surface Model and NDSM - Normalized DSM). Experiments are evaluated on the validation set (area 11, 15, 28, 30 and 40). All quality measures except for 'Overall Acc.' are F1-scores using the ground-truth with eroded boundaries. The best overall accuracy is shown in boldface.

Input image (pixels)	CNN Feature extraction	Window scanning	Total time (sec.)
single-res.	9.8	57.3	67.1
2 × single-res.	18.8	115.4	134.2
3 × single-res.	28.0	171.8	199.8

Table 3: Average evaluation time per image (on the validation set: area 11, 15, 28, 30 and 40) between single-resolution and multi-resolution CNNs.

Multi-resolution CNN feature extraction We employ a multi-resolution deep network that predicts an output based on the 16×16, 32×32 and 64×64 pixel image area. Experimental results are reported in Table 2. For our baseline, we trained a single-resolution network. Table 2 shows an improvement of multi-resolution CNN over a single-resolution CNN based on the average F1-score and overall accuracy. We also report the evaluation time in Table 3. All computations were performed using a single core of an Intel Xeon E5-2680 with 2.70GHz. Both CNN feature extraction and scanning window codes are implemented in `MATLAB` as standard `MATLAB` external calls (MEX-files). Based on Table 2 and 3, we can conclude that the performance gain of multi-resolution CNN is achieved at the cost of increased computation complexity.

Hand-crafted Features and RF Classifier The accuracy of the random forest classifier on the validation set of images is shown in Table 4. The best CNN-only result is included there for convenience. The accuracy is surprisingly high considering the relative simplicity of the hand-crafted features, each using input values from only a single pixel. Table 4 also shows the accuracy of the combined probabilities when used to label the pixels. The overall accuracy improves on the CNN by about 1%, indicating that the hand-crafted features do indeed contain information that is independent from the CNN features.

Conditional Random Field The CRF is applied to the combined CNN and RF probabilities. Inference takes about 1 minute per image on a single CPU. The validation set accuracy is shown in Table 4. Whereas the average pixel accuracy increased only a fraction of a percent, the average F1-score is about 1% higher, the biggest impact being made to the *Car* class. The accuracy is not greatly improved by the CRF, but the aesthetic appeal of the labelling arguably makes it worthwhile. Examples of the effect of CRF smoothing are shown in Figure 5. The main improvements of the CRF are to change the label of regions with ambiguous probabilities, and to remove small mislabelled regions. On the downside, CRF smoothing can sometimes remove small or thin regions.

ISPRS Challenge Test Results The results on the unlabelled test images were submitted to the ISPRS for evaluation. Our results for the 2D labelling challenge are shown in Table 5 for CNN only, combined CNN and RF probabilities, and combined probabilities post-processed with the CRF. In comparison to Table 4 the accuracy is higher than on the validation set, particularly for low vegetation and trees.

Input image resolution (pixels)	Imp. surf.	Building	Low veg.	Tree	Car	Average F1	Overall Acc.
16×16	84.70%	92.15%	72.54%	83.51%	42.54%	75.09%	82.78%
32×32	85.96%	92.42%	74.09%	84.68%	61.06%	79.64%	84.20%
64×64	87.35%	93.34%	74.96%	84.97%	63.32%	80.79%	85.18%
ALL	87.72%	93.27%	75.53%	85.29%	66.89%	81.74%	**85.56%**

Table 2: Performance comparison between single-resolution and multi-resolution CNNs. ALL - We extract CNN features from three different resolutions: 16×16, 32×32 and 64×64 pixels.

	Imp. surf.	Building	Low veg.	Tree	Car	Overall F1	Overall Acc.
Multi-res CNN	87.72%	93.27%	75.53%	85.29%	66.89%	81.74%	85.56%
Random forest	85.83%	92.79%	70.88%	83.98%	0.0%	66.69%	83.47%
CNN+RF	88.58%	94.23%	76.58%	86.29%	67.58%	82.65%	86.52%
CNN+RF+CRF	89.10%	94.30%	77.36%	86.25%	71.91%	83.78%	**86.89%**

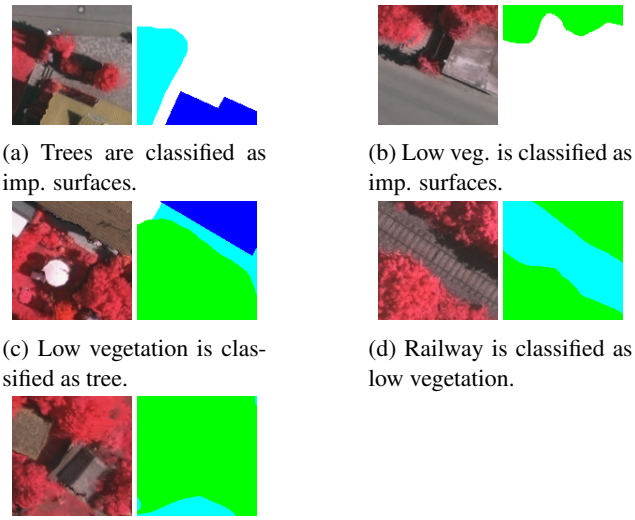
Table 4: Accuracy of RF classifier and CRF labelling on the validation set.

5. Discussion

As was also highlighted in [9] for street scenes, this work demonstrates that CNNs can effectively perform dense semantic labelling of aerial imagery. The features are learned directly from the data rather than being hand-crafted. In contrast to the sophistication and computational cost of the CNN approach, simple pixel-level hand-crafted features achieved almost the same accuracy. Perhaps this is not surprising because the input data are designed to discriminate the target classes: the DSM highlights houses and trees, and infrared highlights vegetation. In single-channel panchromatic images these phenomenologies cannot be relied upon, and the CNN's texture-based approach would be much more accurate.

In [11] it was found that CRF smoothing had a negative effect on accuracy, whereas in our work the accuracy improved. This is most likely because our CRF is defined at the pixel level rather than on super-pixels as in [11]. Along with [11] we conclude that the CRF improves the labelling visually, for example by removing speckle from classifier output labels. Since the CNN is applied with a sliding window, it does not have access to object-level context during classification. CRFs could provide object-level constraints using higher-order cliques or a hierarchical approach, for example to constrain edges of buildings to be straight lines. Labelling of cars could be improved by a rotation-invariant car detector [17]. CRFs provide a probabilistic framework for combining these detections with the classifier labelling [15].

We would like to mention here that the ground-truth provided contains ambiguities, *e.g.*, tree and low vegetation can sometimes be mislabelled. We illustrate some of these mislabelled ground-truths in Fig. 6.



(a) Trees are classified as imp. surfaces.

(b) Low veg. is classified as imp. surfaces.

(c) Low vegetation is classified as tree.

(d) Railway is classified as low vegetation.

(e) Buildings are classified as trees.

Figure 6: Training images and their ground-truth (best view in color). Note that there exist few ambiguities in the provided ground-truths.

References

[1] ISPRS 2D Semantic Labeling Contest. http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html. 1, 5

[2] J. Benediktsson, P. Swain, and O. Ersoy. Neural network approaches versus statistical methods in classification of multisource remote sensing data. *IEEE Trans. on Geoscience and Remote Sensing*, 28(4):540–552, Jul 1990. 2

[3] H. Bischof, W. Schneider, and A. Pinz. Multispectral classification of landsat-images using neural networks. *IEEE Trans. on Geoscience and Remote Sensing*, 30(3):482–490, May 1992. 2

[4] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1124–1137, Sept 2004. 5

[5] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, Nov 2001. 5

[6] S. Decatur. Application of neural networks to terrain classification. In *Int. Joint Conf. on Neural Networks*, volume 1, pages 283–288, 1989. 2

[7] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, 2008. 4

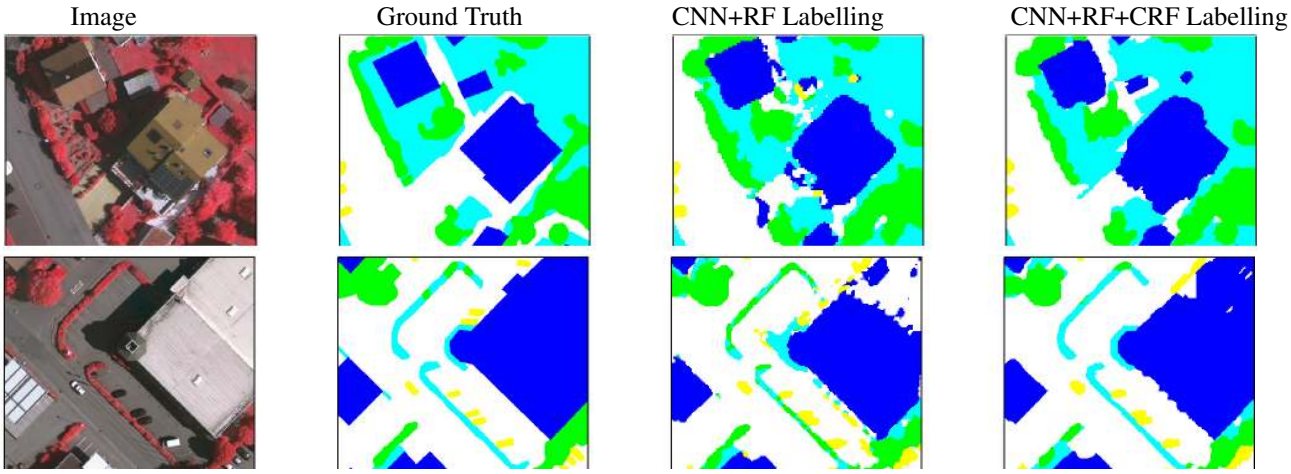


Figure 5: Examples of CRF smoothing. From left to right: input image, ground truth labelling, combined classifier labelling, and CRF labelling.

	Imp. surf.	Building	Low veg.	Tree	Car	Overall Acc.
CNN	88.1%	92.0%	79.0%	86.5%	59.0%	86.1%
CNN+RF	89.0%	93.0%	81.0%	87.8%	59.5%	87.3%
CNN+RF+CRF	89.5%	93.2%	82.3%	88.2%	63.3%	**88.0**%

Table 5: ISPRS 2D Semantic Labeling Contest benchmark results on the hold-out test set. Taken from the challenge web page http://www2.isprs.org/vaihingen-2d-semantic-labeling-contest.html

[8] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Scene parsing with multiscale feature learning, purity trees, and optimal covers. In *Proc. Int. Conf. Mach. Learn.*, 2012. 3

[9] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1915–1929, 2013. 1, 2, 3, 4, 7

[10] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proc. Int. Conf. Mach. Learn.*, 1996. 2

[11] M. Gerke. Use of the stair vision library within the ISPRS 2D semantic labeling benchmark (Vaihingen). Technical report, University of Twente, 2015. 1, 2, 4, 5, 7

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2014. 2

[13] S. Kluckner and H. Bischof. Image-based building classification and 3D modeling with super-pixels. In *Proc. Int. Soc. for Photogrammetry and Remote Sensing, Photogrammetric Computer Vision and Image Analysis*, 2010. 2, 4

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. Adv. Neural Inf. Process. Syst.*, 2012. 2

[15] L. Ladicky, P. Sturgess, K. Alahari, C. Russell, and P. Torr. What, where & how many? combining object detectors and CRFs. In *Proc. Eur. Conf. Comp. Vis.*, 2010. 7

[16] J. Lee, R. Weger, S. Sengupta, and R. Welch. A neural network approach to cloud classification. *IEEE Trans. on Geoscience and Remote Sensing*, 28(5):846–855, Sep 1990. 2

[17] K. Liu, H. Skibbe, T. Schmidt, T. Blein, K. Palme, T. Brox, and O. Ronneberger. Rotation-invariant hog descriptors using fourier analysis in polar and spherical coordinates. *Int. J. Comp. Vis.*, 106(3):342–364, 2014. 7

[18] V. Mnih. *Machine Learning for Aerial Image Labeling*. PhD thesis, University of Toronto, 2013. 1, 2

[19] V. Mnih and G. Hinton. Learning to detect roads in high-resolution aerial images. In *Proc. Eur. Conf. Comp. Vis.*, 2010. 2

[20] T. T. Nguyen, H. Grabner, H. Bischof, and B. Gruber. On-line boosting for car detection from aerial images. In *IEEE Int. Conf. on Research, Innovation and Vision for the Future*, pages 87–95, March 2007. 2

[21] J. Niemeyer, J. Wegner, C. Mallet, F. Rottensteiner, and U. Soergel. Conditional random fields for urban scene classification with full waveform lidar data. In *Photogrammetric Image Analysis*, volume 6952, pages 233–244. Springer Berlin Heidelberg, 2011. 1, 2

[22] J. Porway, K. Wang, B. Yao, and S.-C. Zhu. A hierarchical and contextual model for aerial image understanding. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1–8, June 2008. 1, 2

[23] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2014. 2

[24] R. Rifkin and A. Klautau. In defense of one-vs-all classification. *J. Mach. Learn. Res.*, 5:101–141, 2004. 2

[25] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *Int. J. Comp. Vis.*, 81(1):2–23, 2009. 1, 4

[26] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, 2nd edition, 1999. 2

[27] W. Y. Zou, X. Wang, M. Sun, and Y. Lin. Generic object detection with dense neural patterns and regionlets. *arXiv preprint arXiv:1404.4316*, 2014. 4