Segmentation de scènes extérieures à partir d’ensembles d’étiquettes à granularité et sémantique variables
Damien Fourure, Rémi Emonet, Elisa Fromont, Damien Muselet, Alain Tremeau, Christian Wolf

To cite this version:
Damien Fourure, Rémi Emonet, Elisa Fromont, Damien Muselet, Alain Tremeau, et al.. Segmentation de scènes extérieures à partir d’ensembles d’étiquettes à granularité et sémantique variables. RFIA 2016, Jun 2016, Clermont Ferrand, France. <hal-01318461>

HAL Id: hal-01318461
https://hal.archives-ouvertes.fr/hal-01318461
Submitted on 20 May 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Résumé
Ce papier présente une approche permettant d’utiliser plusieurs bases de données annotées avec différents ensembles d’étiquettes pour améliorer la précision d’un classifieur entrainé sur une tâche de segmentation sémantique de scènes extérieures. Dans ce contexte, la base de données KITTI nous fournit un cas d’utilisation particulièrement pertinent : des sous-ensembles distincts de cette base ont été annotés par plusieurs équipes en utilisant des étiquettes différentes pour chaque sous-ensemble. Notre méthode permet d’entraîner un réseau de neurones convoluti- tionnel (CNN) en utilisant plusieurs bases de données avec des étiquettes possiblement incohérentes. Nous présentons une fonction de perte sélective pour entrainer ce réseau et plusieurs approches de fusion permettant d’exploiter les corrélations entre les différents ensembles d’étiquettes. Le réseau utilise ainsi toutes les données disponibles pour améliorer les performances de classification sur chaque ensemble. Les expériences faites sur les différents sous-ensembles de la base de données KITTI montrent comment chaque proposition contribue à améliorer le classifieur.

Mots Clefs
Deep Learning, Réseaux de neurones de convolution, Segmentation sémantique.

Abstract
In this work, we present an approach that leverages multiple datasets annotated using different classes (different labelsets) to improve the classification accuracy on each individual dataset. We focus on semantic full scene labeling of outdoor scenes. To achieve our goal, we use the KITTI dataset as it illustrates very well the focus of our paper: it has been sparsely labeled by multiple research groups over the past few years but the semantics and the granularity of the labels differ from one set to another. We propose a method to train deep convolutional networks using multiple datasets with potentially inconsistent labelsets and a selective loss function to train it with all the available labeled data while being reliant to inconsistent labelings. Experiments done on all the KITTI dataset’s labeled subsets show that our approach consistently improves the classification accuracy by exploiting the correlations across datasets both at the feature level and at the label level.

Keywords
Deep Learning, Convolutional neural network, semantic labelling, inconsistent labelling.

1 Introduction
Semantic scene parsing (a.k.a. semantic full scene labeling) from RGB images aims at segmenting an image into semantically meaningful regions, i.e. to provide a semantic class label for each pixel of an image — see Table 2 for examples of labels in an outdoor context. Semantic scene parsing is useful for a wide range of applications, for instance autonomous vehicles, automatic understanding and indexing of video databases, etc.

Most semantic scene parsing methods use supervised machine learning algorithms and thus rely on densely labeled (manually annotated) training sets which are very tedious to obtain. Only a small amount of training data is currently available for this task, which makes this problem stand out from other problems in vision (as for instance object recognition and localization). This is a particularly stringent problem for the currently most performing family of models in computer vision, namely deep networks, which are particularly needy in terms of training data. In the case of depth images, data-augmentation using artificially-created training data has been employed successfully for segmentation problems [1, 2]. However, the high variations of content in fully textured images make this solution at the moment very difficult to use for RGB images.

Most datasets for scene parsing contain only several hundreds of images, some of them only several dozen [3, 4, 5, 6, 7, 8, 9, 10, 11]. Combining these datasets is a non-trivial task as target classes are often tailored to a custom application. For example, one might be interested in specific types of vegetation like trees, bushes and grass, or types of objects such as graffiti or billboard while other applications do not require discriminating between these types. In this work, we present a solution to this problem by learning a
joint prediction model on a dataset composed of different and inconsistently labeled subsets.

The contributions of this paper are threefold: (i) we propose a deep network capable of processing features and labels (as input) from several different datasets and providing predictions for each of these datasets. Early layers are shared over datasets, whereas later layers are separated; (ii) we show that shared parameters can lead to significant improvements of performance even if the semantic meanings of the different labelsets are different; (iii) our experiments done on 7 annotated subsets of the KITTI dataset [12, 13] show that the label definitions, albeit inconsistent, are highly correlated, and that the proposed networks are able to capture and exploit these correlations.

2 Related Works

In this section, first, we discuss the state of the art on semantic scene parsing, then, we focus on feature and knowledge transfer methods, especially in the context of deep networks. Finally, we report several specific issues related to the KITTI benchmark which sparked wide interest and for which a limited range of methods has been proposed, especially in scene parsing.

2.1 Semantic Segmentation

Whereas the methods used for low level segmentation are diverse, high level semantic segmentation is dominated by machine learning, which can be explained by the high intra-class variances of this task. Learning methods range from random forests, to Support Vector Machines and deep networks. In [14] for instance, a structured-output version of random forests is proposed, where each leaf node predicts labels for every single pixel of an input patch, and predictions are integrated over patches using voting. Deep neural networks have also been used in wide range of works [15, 16, 17].

Over years, segmentation algorithms (semantic or not) have often been regularized through probabilistic graphical models like Markov Random Fields or Conditional Random Fields (CRF). Inference in these models requires to solve combinatorial problems which are often non-submodular and intractable. These methods have also been combined with machine learning, in particular deep networks [15]. Regrouping pixels into larger structures, like super-pixels, is also a frequently used technique [18, 15].

Auto-context models [19] are a different way to include structural information, which is computationally less expensive. They are defined by cascades of predictors, each one improving on the result of the previous. These models have also been adapted for scene parsing. In [16], similarly to recurrent networks for sequence classification, the same network is applied multiple times to outputs of different stages. In [17], a context learner is trained to predict context for a subsequent refinement learner. Both networks are trained on segmentation maps, but the refinement learner is trained to cope with noisy segmentations as context input. In [20], scene parsing from depth images using auto-context is formulated as a graphical model and solved through message-passing.

Recent methods try to tackle the problem in a weakly supervised setting alleviating the problem on manual annotations [21, 22]. Instead of requiring pixelwise ground-truth, they integrate imagewise information, or pointwise ground-truth which can be easily provided. They are usually strongly regularized through priors like objectness [21] or classification performance based on the full image [22].

Segmentation methods specifically developed for the KITTI dataset are described in subsection 2.3.

2.2 Transferability

Lots of recent papers [23, 24, 25, 26, 27] proposed methods to solve the problem of the transferability of deep features. Since CNN require a lot of labeled data to provide very good features, the trend consists in exploiting features learned on one big dataset and in adapting them to other datasets and other tasks [26]. For example, in an extensive analysis about deep feature transfer, Yosinski et al. [26] show that it is better to transfer lower layers features learned on a different (and maybe distant) task than using random weights. These transferred features improve generalization performance even after fine-tuning on a new task. Hinton et al. [25] propose another way to transfer (or distill) knowledge from one big network to a small one. The idea is for the small network to learn both the outputs (soft targets) of the big network as well as the correct labels of the data. Accounting for the soft labels from the other network helps in learning the correlation between the labels. Furthermore, the authors showed that this distillation works well even when the transfer set that is used to train the final small model lacks any samples of one or more of the classes. When the dataset used to learn the first (source) network is different from the dataset used to fine-tune the final (target) network, it can be interesting to force the target and source features to be similar. This is the idea of Ganin and Lempitsky [23] who learn features that are invariant with respect to the shift between the source and target domains. Their solution consists in learning a domain classifier that tries to discriminate the two domains and to reverse the gradient during the backpropagation in order to learn features that cannot help in discriminating the domains. Very recently, Tzeng et al. [24] merge the two previous ideas (soft labels and domain confusion) into a framework that allows to transfer network knowledge across domains and tasks.

Still in the context of domain adaptation, Zhang et al. [27] propose to match the source and target marginal distributions of features as well as the source and target conditional distributions of the labels associated to the features. Therefore, while learning the target network, they minimize both the marginal and conditional empirical Maximum Mean Discrepancy (MMD) between the source and target distributions. The domain adaptation problem is a bit different from our current problem, where we have a single dataset
partially labeled by different authors, with different and inconsistent labelings. Nevertheless, we can report three important points from the papers listed above: i) increasing the data helps generalization; ii) fine-tuning for each specific task improves the classification and iii) exploiting the correlations between the labels also helps the classification. These observations guide the proposed approach.

3 Proposed Approach

Problem statement. Given a set of images to label, which are drawn from a set of $K$ different datasets, the pairs of input patches $x_i^k$ and output labels $y_i^k$ are grouped into sets $D_k = \{x_i^k, y_i^k\}$, where $k=1\ldots K$ and $i$ indexes pixels. The label spaces are different over the different datasets, therefore each $y_i^k$ can take values in space $\mathcal{L}^k$. Our goal is to learn a nonlinear mapping $y = \theta(x, \Theta)$ with parameters $\Theta$ which minimizes a chosen risk $R[\theta(x, \Theta) \neq y]$. The mapping $\theta$ is represented as a convolutional neural network, where each layer itself is a nonlinear mapping $f_i(W_i, h_{i-1} + b_i)$ where $h_i$ is the $l^{th}$ hidden representation, $W_i$ and $b_i$ are the weights and bias of the $l^{th}$ layer and $f(.)$ is the activation function of the $l^{th}$ layer.

We propose to minimize the empirical risk, then $R[\theta(x, \Theta) \neq y] = \frac{1}{N} \sum_{k=1}^{N} J(x, y, \Theta)$, where $N$ is the number of training samples and $J$ is the loss function for each individual sample. We also propose to use the cross entropy loss $J(x, y, \Theta) = -\sum_{j} 1_{y=j} \log \theta(x, \Theta)_j$, where $\theta(x, \Theta)_j$ is the network output for class $j$.

Limitations of separate training. Considering $K$ different datasets, the classical baseline approach is to train $K$ separate mappings (models) $\theta^k$, each defined on its own label set $\mathcal{L}^k$. Unfortunately this basic approach (illustrated in Figure 1a) presents several shortcomings:

(i) Each mapping $\theta^k$ is trained on its own dataset $D^k$, requiring a minimization over a separate sets of parameters $\Theta^k$.

In the chosen deep convolutional implementation the parameters $\Theta^k=\{W^k, b^k\}_{j=1}^L$ including all convolution filters and the weights of all fully connected layers, which are generally large sets ($< 2$ millions parameters). Learning such a large amount of parameters from limited (and generally small) amounts of training data is very challenging.

(ii) Relationships between label spaces are not modeled, which further limits the power of the trained models.

Joint feature training with selective loss. We propose to tackle shortcoming (i) by exploiting the hierarchical nature of deep models. It is well known that, on most classical problems in computer vision, supervised training leads to a rising complexity of features over layers. Meanwhile early layers extract low level features, which exhibit strong independence of training input distribution and even task, later layers extract features which are more and more specific to the problem at hand [29]. We also propose to train a single deep network on the union of all individual datasets. Our network will share the parameters of its earlier layers for all datasets (to reduce overfitting), whereas the later layers will be duplicated for each dataset. This joint training approach is illustrated in Figure 1b. There is one output unit per label in the union of all label sets $\mathcal{L}$, which means that the output layer is able to provide predictions for each of the handled datasets. In a traditional multi-class setting involving a probabilistic loss function such as the cross-entropy, the network output $p_j$ is computed using a soft-max function in the last layer. Classically, this can be seen as minimizing the negative log-likelihood (NLL) of the ground truth classes. However, with $K$ different datasets, this choice is counter-productive as it also maximizes the NLL of all other classes, including classes that are not used in the training set. This will
lead to problems when there exists a correlation between labels across different datasets. As concrete example, in the KITTI dataset (see Table 2 where all labels are reported) the class Tree of the dataset from He et al. [5] is likely to be correlated with the class Vegetation from the dataset labeled by Kundu et al [6]. In our model, the normalization of the output probabilities is achieved using a dataset-wise soft-max : for each label \( j \) from dataset \( k \),

\[
f(j, \theta(x, \Theta)) = \frac{e^{\theta(x, \Theta)j}}{\sum_{j' \in \mathcal{L}_k} e^{\theta(x, \Theta)j'}}
\]

(1)

In practice, the datasetwise soft-max is combined with a selective cross-entropy loss function as follows :

\[
J'(k, x, y, \Theta) = -\theta(x, \Theta)y + \log( \sum_{j \in \mathcal{L}_k} e^{\theta(x, \Theta)j})
\]

(2)

Gradients are null for parameters involving output units corresponding to labels from datasets \( l \) with \( l \neq k \). This is equivalent to using separate output layers for each dataset and intermediate layers with shared parameters over the different datasets.

**Modeling correlations between labelsets.** Shortcoming (ii) is partly addressed by the joint feature training, as correlations between labels across datasets can be learned by the shared layers in the network. On the other hand, we will show that explicitly modeling these correlations further improves the discriminative power of the classifier. To take into account the correlation between labelsets, we concatenate the outputs of the different networks \( \theta^k(x, \Theta) \). We then feed the concatenation into a datasetwise log-soft-max (see Eq. 1) followed by some additional fully-connected layers. After pre-training each individual \( \theta^k(x, \Theta) \), new mappings \( \theta^k(x, \Theta) \) are trained, where each \( \theta^k \) combines the inputs from all \( \theta^k, k = \{1 \ldots K\} \). In an end-to-end training setting, parameters of the full model are trained jointly (see Figure 1d).

We do not try to estimate from which dataset an input sample actually originated. Unlike transfer learning tasks, no shift is supposed in the input distributions of the different datasets, the difference being in the ground truth labels. Existing shifts in the input distributions can still be learned by the network, however.

### 4 Experimental Results

#### 4.1 Training details

All experiments are done with the Torch7 [30] framework and training was operated on consumer GPUs. We used a
network architecture inspired by Farabet et al.[15] for outdoor scene labeling and improved with the recent advances in deep neural network research. Our network is composed by 3 convolutional layers followed by 2 fully-connected linear layers. The network is illustrated in Figure 1. The first two convolutional layers are composed by a bank of 7 × 7 filters followed by ReLU [31] units, 2×2 maximum pooling and batch normalization [32] units. The last convolutional layer is a simple filter bank followed by a ReLU unit, a batch normalization unit and dropout [33] with a drop factor of 30%. The first fully connected linear layer is then followed by a ReLU unit and the last layer is followed by our dataset-wise softmax unit. We used this network to the our approach to obtain the results shown in lines 3 and 4 of Table 3 (illustrated in Figure 1c and 1d). Additional fully connected layers are added followed by a ReLU unit (and the dataset-wise softmax unit for the output layer) and batch normalization units. To train the network, RGB images are transformed into YUV space. A training input example is composed of a patch \( x_i \) of size 46 × 46 cropped from the original image and centered around a pixel \( i \), the dataset \( k \) from which the example comes from, and the label \( y_k^i \) corresponding to pixel \( i \). Stochastic gradient descent with a mini-batch of size 128 was used to update the parameters. We used early stopping on a validation set in order to stop the training step before overfitting.

### 4.2 Dataset details

In addition to the information provided in Section 2.3 about the KITTI dataset, we now provide precise information about the labeled data. This is summarized in Table 1. The dataset has been partially labeled by seven research groups resulting in 736 labeled images that are split into : a train set, a validation set and a test set. When the information was given by the author, we used the same train/test set as them. Note that Xu et al. [10] provide a complete hierarchy of very detailed labels, but we only used the highest level of the hierarchy to obtain a labeling more compatible (in terms of granularity) with the labels from the other research teams. Also note that the KITTI dataset contains over 40000 frames (180GB of raw videos) but in this work we only rely on the labeled data. We then sample on average 390.000 patches in each video frame (depending on its size). This results into a dataset of about 280 million patches suitable to train a deep learning architecture.

As mentioned in Section 2.3, the different labels provided by the different teams are not always consistent. As illustrated in Table 2, we can see that the granularity and the semantics of the labels may be very different from one labeling to another. For example, Ladicky et al. separate the Trees from the Grass. However, this might correspond to the Vegetation labels in the subset from Xu et al. but might also correspond (in the case of Grass) to the labels Ground. He et al. [5] have not used the labels Pole, Sign or Bike in most other labelings. These labels are likely to overlap with the label Building of He et al. but then, this Building class cannot be consistent anymore with the other labelings that contain the label Building in other subsets. Some groups have used the label Bike and some others have used the label Cyclist. Those two labels are likely to overlap but in one case a team has focused on the entire entity “cyclist on a bike” whereas another has only focused on the bike device. These inconsistencies made the KITTI dataset a good candidate to test the proposed method.

### 4.3 Joint training

Table 3 shows the accuracy obtained for all our training strategies. We report the global accuracy and the average accuracy. Global is the number of correctly classified pixels over the total number of pixels (also called recall or pixel accuracy) and the Average is the average of this recall per class (also called the class accuracy). Note that the last column (Total) of the Table 3 gives the global and average accuracies for all sub datasets together so that the totals take into account the relative number of labeled pixels in each sub-dataset instead of being the average of all elements in the line.

The first learning strategy implemented consists in learning one network per dataset with the architecture described in Section 4 and illustrated in Figure 1.a. This is our baseline. The results for this strategy are shown in the first line (No Fusion) of Table 3. Note that the state-of-the-art results for each of those sub datasets are (respectively for global and average accuracies) : (92.77, 68.65) for He et al. [5] ; (97.20, non reported) for Kundu et al. [6] ; (82.4, 72.2) for Ladicky et al. [7] ; (51.2, 61.6) for Ros et al. [8] ; (90.9, 92.10) for Sengupta et al. [9] ; (non reported, 61.6) for Xu et al. [10] ; and (89.3, 65.4) for Zhang et al. [11]. These results are often much better (except for Ros et al.) than those reported in Table 3. This can be obviously explained by the fact that : [6, 9, 7, 8] only show results computed from a subset of their labels (e.g. the label pedestrian is ignored in [9, 7, 8]) ; the features used by all methods are richer (e.g. depth and time) as discussed in Section 2.3 ; and the proposed methods always combine multiple classifiers tuned on one particular sub-dataset.
The second alternative strategy (Joint Training) consists in learning one single network (illustrated in Figure 1b) with all the datasets using the selective loss function detailed in Section 3. The second line of Table 3 shows that this strategy gives better results than our baseline (in total +1.81 for the pixel accuracy and +3.85 for the class accuracy). This result was expected since the amount of data used clearly increases from the No Fusion network (with a maximum of 112 training images for the subset labeled by Zhang et al.) to the Joint training strategy which uses 368 training images. This data augmentation leads to a better generalization and so to better classification accuracies. These results show that the selective loss function allows to successfully take into account the inconsistent labels.

4.4 Correlation rates between labelsets

When the Joint training network is trained, the correlation rates between labels of each sub dataset can be computed from the output probability vector of the data set-wise softmax units. We computed a label correlation matrix, shown in Figure 3, by averaging the predictions made by the network for each target class label (from one of the 7 possible labelings). The (full) diagonal of the matrix gives the correlation rates between the expected target labels. In each line, the other non-zero values correspond to the labels correlated with the target label $y^i_k$. Please note that this is different from a confusion matrix since each line shows the average of the output probabilities and not the predictions. We can also see that a diagonal is visible in each block of the matrix (corresponding to the correlations between different labelings) for the first 5 labels of each dataset. This means that, as expected, these first 5 labels are all correlated (for example, the label Road from He et al. is correlated with the label Road from Kundu et al. with the Road from Ladicky et al. etc.). A second observation is that the correlation matrix is not symmetric. For example, the classes Building, Poles, Signage and Fence from Xu et al. have (as already discussed in Section 4.2) a high correlation with the class Infrastructure from Xu et al., meaning that these classes overlap. On the contrary, the class Infrastructure from Xu et al. has a very high correlation with the class Building from Sengupta et al. and a limited one with the classes Poles, Signage and Fence. This is due to the target distributions. The Building class from Sengupta et al. is more represented than the three other classes so Infrastructure from Xu et al. is more correlated to Building. If these observations mostly confirm the expectations we discussed in Section 4.2, they show that our method can also be used to automatically discover correlations between labels.

4.5 Improvement from correlation modeling

The correlations studied in Section 4.4 not only help us to retrieve the hierarchical dependencies between all the labels but it can also improve the prediction accuracies. Using our method, the network can correct possible errors if another prediction correlated to the target one is more confident. These correlations are taken into account by adding a Multi-Layer Perceptron (MLP) after the output of the original network and by fine tuning the entire network with this added MLP.

The Joint training with shared context strategy, illustrated in 1c, consists in learning all datasets together with our selective loss-function (as for the Joint training). This third approach improves pixel accuracy (global) but not class accuracy (average), compared to Joint training (see Table 3 line 3 vs line 2). As the stochastic gradient descent optimizes the pixel accuracy (global) this leads to an overfitting. Indeed, an increase in average can be a sign that the network starts to over-specialize: it strongly learns the distribution of classes to the detriment of learning the appearance of each class.
<table>
<thead>
<tr>
<th>No Fusion</th>
<th>Global</th>
<th>Average</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>He</td>
<td>74.67</td>
<td>58.56</td>
<td>80.94</td>
</tr>
<tr>
<td>Kundu</td>
<td>72.48</td>
<td>56.04</td>
<td></td>
</tr>
<tr>
<td>Ladicky</td>
<td>72.94</td>
<td>43.16</td>
<td></td>
</tr>
<tr>
<td>Ros</td>
<td>76.96</td>
<td>48.76</td>
<td></td>
</tr>
<tr>
<td>Sengupta</td>
<td>78.71</td>
<td>71.26</td>
<td></td>
</tr>
<tr>
<td>Xu</td>
<td>89.67</td>
<td>83.11</td>
<td></td>
</tr>
<tr>
<td>Zhang</td>
<td>84.98</td>
<td>57.39</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Joint training**

<table>
<thead>
<tr>
<th>Global</th>
<th>Average</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>78.68</td>
<td>64.41</td>
<td>82.75</td>
</tr>
<tr>
<td>77.20</td>
<td><strong>60.61</strong></td>
<td>(+1.81)</td>
</tr>
<tr>
<td>75.86</td>
<td>46.52</td>
<td></td>
</tr>
<tr>
<td>78.22</td>
<td>52.06</td>
<td></td>
</tr>
<tr>
<td>81.48</td>
<td>75.64</td>
<td></td>
</tr>
<tr>
<td>88.02</td>
<td><strong>85.14</strong></td>
<td>(+3.85)</td>
</tr>
<tr>
<td>86.89</td>
<td>57.39</td>
<td></td>
</tr>
<tr>
<td>88.69</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Joint training with shared context**

<table>
<thead>
<tr>
<th>Global</th>
<th>Average</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>78.61</td>
<td>62.87</td>
<td>83.16</td>
</tr>
<tr>
<td>77.76</td>
<td>59.13</td>
<td>(+2.22)</td>
</tr>
<tr>
<td>76.00</td>
<td>45.22</td>
<td></td>
</tr>
<tr>
<td>78.40</td>
<td>51.16</td>
<td></td>
</tr>
<tr>
<td>81.97</td>
<td>75.55</td>
<td></td>
</tr>
<tr>
<td>88.43</td>
<td>84.94</td>
<td></td>
</tr>
<tr>
<td>87.54</td>
<td>59.75</td>
<td></td>
</tr>
<tr>
<td>83.16</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Joint training with individual context**

<table>
<thead>
<tr>
<th>Global</th>
<th>Average</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>79.31</td>
<td>64.15</td>
<td><strong>83.19</strong></td>
</tr>
<tr>
<td>77.53</td>
<td>59.77</td>
<td>(+2.25)</td>
</tr>
<tr>
<td>76.81</td>
<td>47.92</td>
<td></td>
</tr>
<tr>
<td>78.41</td>
<td><strong>52.35</strong></td>
<td>(+4.10)</td>
</tr>
<tr>
<td>80.98</td>
<td>58.09</td>
<td></td>
</tr>
<tr>
<td>88.35</td>
<td>59.84</td>
<td></td>
</tr>
<tr>
<td>86.76</td>
<td>59.75</td>
<td></td>
</tr>
<tr>
<td>83.19</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3 – Pixel (Global) and Class (Average) accuracy results for the 7 used sub-datasets with 4 different training strategies: NF=No Fusion (see Fig. 1a); JT= Joint training (see Fig. 1b); JTSC=Joint training with shared context (see Fig. 1c); JTIC=Joint training with individual context (see Fig. 1d). Best results are highlighted in bold.

The Joint training with individual context (JTIC), illustrated in 1d, is based on the same approach as the Joint Training (JT) with shared context, but instead of learning all datasets together with the selective loss function the datasets are here learned one by one (by fine-tuning one network per dataset). The shortcoming of the JT approach is that when all datasets are learned together the features are general. By fine-tuning with only one dataset, we specialized the features for this datasets. The results from Table 3 line 4 show that in average this approach outperforms the other ones. Both the global and the average measures increase compared to Joint training, suggesting that this approach is more robust that the Joint training with shared context.

The confusion matrices computed from our baseline strategy next from the JTIC are shown in Figure 2. In order to better visualize the improvement, the matrix on the right shows the difference between the JTIC confusion matrix and the baseline one. Negative values are displayed in red and positive values in blue. As expected the majority of non-diagonal values are negative while the diagonal values are positive. This demonstrates that errors (non-diagonal) made by the first network were reduced by better (i.e. correct) predictions (diagonal) with JTIC. Looking more into the details, we can see what errors are corrected by JTIC. In most datasets, an important amount of cars were labeled as buildings by the baseline, meanwhile now they are properly labeled with our JTIC approach. To a lesser extent, the same is also observed for sidewalks and grounds.

5 Conclusion

This paper proposed and evaluated different strategies to take into account inconsistent labels in correlated data to improve outdoor full scene labeling. The proposed methods have been applied to the particular case of the KITTI dataset using only the RGB color features and labeled data of this dataset. To tackle the full scene labeling problem, one could rely on much more detailed features such as depth information, temporal information and the huge amount of unlabeled data available for this dataset. This constitute our future work. The main contribution of this paper is to propose a Joint training with individual context approach that allows to : leverage multiple datasets annotated using different classes (different labelsets); and improve the accuracy of the classification on each individual dataset. Experimental results show a clear improvement in terms of classification accuracy when multiple datasets are fused and further improvement when the last layers of the network are computed datasetwise.

Acknowledgment

Authors acknowledge the support from the ANR project SoLSStiCe (ANR-13-BS02-0002-01).
Références


