Semantic Segmentation via Multi-task, Multi-domain Learning

Damien Fourure[1], Rémi Emonet[1], Elisa Fromont[1(✉)], Damien Muselet[1], Alain Trémeau[1], and Christian Wolf[2]

[1] Universite de Lyon, UJM, CNRS, Lab Hubert Curien UMR5516, F-42000 Lyon, France
elisa.fromont@univ-st-etienne.fr
[2] Universite de Lyon, CNRS, INSA-Lyon, LIRIS, UMR5205, F-69621 Lyon, France

Abstract. We present an approach that leverages multiple datasets possibly annotated using different classes to improve the semantic segmentation accuracy on each individual dataset. We propose a new *selective loss* function that can be integrated into deep networks to exploit training data coming from multiple datasets with possibly different tasks (e.g., different label-sets). We show how the gradient-reversal approach for domain adaptation can be used in this setup. Thorought experiments on semantic segmentation applications show the relevance of our approach.

Keywords: Deep learning · Convolutional neural networks · Semantic segmentation · Domain adaptation · Multi-task learning

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 Fig. 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 deep network models who are particularly needy in terms of training data.

Most datasets for scene parsing contain only several hundreds of images, some of them only several dozen [6,9,13–15,18,19,22,25]. Additionally, combining these datasets is a non-trivial task as target classes are often tailored to a custom application. One good example of such label-set diversity can be found within the KITTI Vision benchmark [4] which contains outdoor scene videos. Many research teams work on this dataset since its release in 2013, tackling computer vision tasks such as visual odometry, 3D object detection and 3D tracking

A. Robles-Kelly et al. (Eds.): S+SSPR 2016, LNCS 10029, pp. 333–343, 2016.
DOI: 10.1007/978-3-319-49055-7_30

[9,13,14,18,19,22,25]. To tackle these tasks, several research teams have labeled parts of the original dataset, independently from the other teams and often for different goals (among the works listed above, semantic segmentation is the final goal only for [14,25]). In practice, the ground truth segmentation quality varies and both the granularity and the semantics of the labels differ. This inconsistency in the label-set is also true when using the Stanford Background [6] and SIFT Flow [15] datasets in combination with or in addition to KITTI.

The contributions of this paper are threefold: (I) we formalize a simple yet effective *selective loss* function that can be used in shared deep network to exploit training data coming from multiple datasets having different label-sets (different segmentation tasks). (II) we show that the gradient-reversal approach for domain adaptation [3] can be used but needs to be manipulated with care especially when datasets are unbalanced, (III) we run thorough scene segmentation experiments on 9 heterogeneously labeled datasets, underlining the impact of each part of the approach.

2 Related Works

In this section, we first discuss state-of-the-art methods dealing with multiple datasets, focusing in particular on feature and knowledge transfer methods in the context of deep networks. We then discuss semantic scene parsing with an emphasis on methods used for the KITTI benchmark.

Learning Across Datasets. Many recent papers [3,7,21,23,26] proposed methods to solve the problem of the transferability of deep features. Since CNNs require a lot of labeled data to provide good features, the usual methods exploit features learned on one big dataset and adapt them to other datasets and tasks [23]. In an extensive analysis about deep feature transfer, Yosinski et al. [23] show that it is better to initialize lower layers from 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. [7] propose another way to transfer (or distill) knowledge from one large network to a smaller one. The idea is for the small network to learn both the outputs of the large network as well as the correct labels of the data.

Considering the same task but datasets in multiple domains (different distributions of input data), the theory of domain adaptation tells us that the most similar the features are across domains [1,5,16], the better the adaptation will be. Ganin and Lempitsky [3] follow this principle by learning features that are invariant with respect to the shift between the source and target domains. In order to do that, they train a *domain classifier* and reverse the gradient during the backpropagation step in order to learn features that cannot help in discriminating the domains.

In this paper we show how domain adaptation techniques can be integrated in our proposed method to benefit from multiple heterogeneously labeled datasets without compromising the classification accuracy on each dataset.

Semantic Segmentation. Whereas the methods used for low level segmentation are diverse, high level semantic segmentation is dominated by machine learning. Learning methods range from random forests, to Support Vector Machines and deep learning who have been used in wide range of works [2,11,17]. Over years, segmentation algorithms have often been regularized through probabilistic graphical models like Conditional Random Fields (CRF). These methods have also been combined with deep networks [2,20]. For the 7 subsets of the KITTI dataset used in this paper [9,13,14,18,19,22,25], deep learning has never been used to tackle the semantic segmentation step. For example, [14] shows how to jointly classify pixels and predict their depth using a multi-class decision stumps-based boosted classifier. [25] uses a random forest classifier to classify segments of an image for different scales and sets of features. The aim of this paper is to show how to learn across datasets (with possibly different tasks) to improve the classification results. In particular, we do not optimize our method to produce the best accuracy for each of the used dataset. For example, while in KITTI many authors use rich features such as color, depth and temporal features, and complex post processing, we only use the RGB channels for our experiments and we do not take the label hierarchies into account. We also do not use any post-processing step.

3 Proposed Approach

Problem Statement. Given a set of images drawn from a set of K different datasets, pairs made of an input patch x_i^k and a target label y_i^k are grouped into sets $D_k = \{x_i^k, y_i^k\}$, where $k{=}1 \dots K$ and i indexes patches. The label spaces are different over the 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 Θ which minimizes a empirical risk $\mathcal{R}[\Theta, D]$. The mapping θ is represented as a convolutional neural network, where each layer itself is a nonlinear mapping $f_l(\mathbf{W}_l \mathbf{h}_{l-1} + \mathbf{b}_l)$ where $\mathbf{h}_l$ is the l^{th} hidden representation, $\mathbf{W}_l$ and $\mathbf{b}_l$ are the weights and bias of the l^{th} layer and $f_l(.)$ is the activation function of the l^{th} layer. We minimize the empirical risk, $\mathcal{R}[\Theta, D] = \frac{1}{N} \sum_{k=1}^{K} \sum_i J(x_i^k, y_i^k, \Theta)$, where N is the number of training samples and J is the loss function for each individual sample. We use the cross entropy loss $J(x_i^k, y_i^k, \Theta) = -\log \theta(x_i^k, \Theta)_{y_i^k}$, where $\theta(x_i^k, \Theta)_{y_i^k}$ is the network output for class y_i^k.

Limitations of Separate Training. Considering K different datasets, the classical baseline approach is to train K separate mappings (models) θ^k, each defined on its own label set $\mathcal{L}^k$. This baseline approach is illustrated in Fig. 1a. Unfortunately this basic approach presents the shortcoming that each mapping θ^k is trained on its own dataset D^k, which requires minimizing over separate sets of parameters Θ^k. In deep convolutional networks, the parameters Θ^k include all convolution's filters and the weights of all fully connected layers (overall, several millions of parameters). Learning such a large amount of parameters from limited (and generally small) amounts of training data is very challenging.

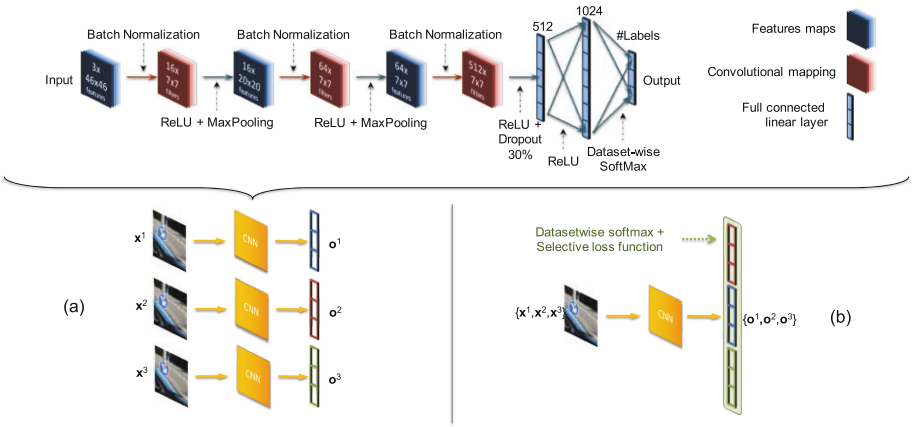


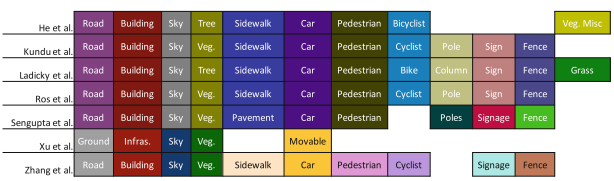
Fig. 1. Network used for our experiments with different learning strategies (**(a)** and **(b)**). **(a)** *No Fusion* is our baseline and consists in learning one network per dataset. **(b)** *Joint training* consists in learning a single network with our selective loss function.

Joint Feature Training with Selective Loss. We propose to tackle this shortcoming by exploiting the hierarchical nature of deep models. On most classical problems in computer vision, supervised training leads to a rising complexity and specificity of the features over the layers [24]. In our case, we propose to train a single deep network on the union of all individual datasets. This allows the network to decide at every layer which features should be generic and which ones should be task-specific. This joint training approach is illustrated in Fig. 1b. There is one output unit per label in the union of all label sets $\mathcal{L}^k$: the output layer thus produces predictions for all considered datasets.

In a traditional multi-class setting, the network output is computed using a soft-max function to produce a probability vector. However, with K different datasets, this is counter-productive: it maximizes the target class probability but minimizes the probability of all other classes, including the ones from different label sets. This minimization is problematic when there exists a correlation between labels across different datasets. For example, in the KITTI dataset (see Fig. 2 where all labels are reported) the class *Tree* of the dataset from He et al. [9] is correlated with the class *Vegetation* from the dataset labeled by Kundu et al. [13]. A plain softmax, optimizing the probability of the *Tree* class will implicitly penalize the probability of *Vegetation*, which is not a desired effect.

We thus define the *dataset-wise soft-max* (that produces a probabilities vector per dataset): for an input sample x_i^k, y_i^k from dataset k (denoted x, y for readability),

$$f(\theta(x, \Theta), y, k) = \frac{e^{\theta(x,\Theta)_y}}{\sum_{j' \in \mathcal{L}^k} e^{\theta(x,\Theta)_{j'}}} \tag{1}$$



Data	Train	Val	Test	Total
He [9]	32	7	12	51
Ku [13]	28	7	15	50
La [14]	24	6	30	60
Ro [18]	80	20	46	146
Se [19]	36	9	25	70
Xu [22]	56	14	37	107
Zh [25]	112	28	112	252
Total	368	91	277	736

Fig. 2. The 68 labels (with the original colors) used by the different authors to annotate their subset of the KITTI benchmark as well as the number of images (and their train/validation/split decomposition, see details in Sect. 4) in each subset. (Color figure online)

During learning, the dataset-wise soft-max is combined with a *selective cross-entropy loss* function as follows:

$$J'(\theta(x,\Theta),y,k) = -\theta(x,\Theta)_y + \log(\sum_{j' \in \mathcal{L}^k} e^{\theta(x,\Theta)_{j'}}) \tag{2}$$

Gradients are null for parameters involving output units corresponding to labels from datasets l with $l \neq k$. This is equivalent to having a separate output layer for each dataset and intermediate layers that are shared across datasets.

Gradient Reversal for Domain Adaptation. So far, our method trains a single network on several datasets adapting for different label-sets, while ignoring potential shift in the input domain between datasets. This is not a problem in the case where the input data is sampled from a single distribution (e.g. for the different subsets of KITTI). In other cases, a non neglectable shift in input distribution does exist, for instance between the Stanford and SiftFlow data.

The theory of domain adaptation tells us that a better adaptation between source and target domains can be achieved when the feature distributions of both sets are closer to each other [1,5,16]. In the lines of Ganin and Lempitsky [3], this can be achieved using an additional classifier trained on the same features, which attempts to predict the domain of the input data. In the case of domain invariant features, this classifier will achieve high error. More precisely, our full mapping $y = \theta(x,\Theta)$ is conceptually split into two parts: the *feature extractor* $f = \theta_f(x,\Theta_f)$, which corresponds to the first convolutional layers and results in features f, and the *task classifier* $y = \theta_t(f,\Theta_t)$, which corresponds to the later fully connected layers including the selective loss function. The *domain classifier* is an additional mapping, which maps the features f to an estimated domain d, given as $d = \theta_d(f,\Theta_d)$ The goal here is to *minimize* the loss of the domain classifier θ_d over its parameters Θ_d in order to train a meaningful classifier, and to *maximize* the same loss over the features f, i.e. over the parameters Θ_f of the feature extractor θ_f, in order to create domain invariant features. In the lines of [3] this is achieved through a single backpropagation pass over the full network implementing θ_f, θ_t and θ_d, inversing the gradients between the domain classifier θ_d and the feature extractor θ_f. In practice, the gradients are not only inversed,

they are multiplied with a hyper-parameter $-\lambda$, which controls the importance of the task classifier and the domain one, as in [3].

Our experiments described in Sect. 4 show, that this domain adaptation step is also useful and important in our more general setting where results are requested for different tasks.

4 Experimental Results

Training Details. For all experiments we used a network architecture illustrated at the top of Fig. 1. This architecture is a modified version of Farabet et al. [2]. The first two convolutional layers are composed by a bank of filters of size 7×7 followed by ReLU [12] units, 2×2 maximum pooling and batch normalization [10] units. The last convolutional layer is a filter bank followed by a ReLU unit, a batch normalization unit and dropout [8] 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. To train the network, each RGB image is transformed into YUV space. A training input example is composed of a patch x_i of size 46×46 cropped from an image, the dataset k from which the image comes from, and y_i^k, the label of the center pixel of the patch x_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 over-fitting. The only data augmentation strategy that we used is an horizontal flip of patches.

Datasets Details. The KITTI dataset [4] has been partially labeled by seven research groups [9,13,14,18,19,22,25] resulting in 736 labeled images (with almost no images in common) 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, otherwise we randomly split them into approximately 70 % of data for the training and validation set and 30 % data for the test set, ensuring that any two images from the same video sequence end up in the same split. The labels used in the different subsets of the KITTI dataset are summarized in Fig. 2. We 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 Sect. 2, the different labels provided by the different teams are not always consistent. As illustrated in Fig. 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. have not used the labels *Pole*, *Sign* or *Fence* used 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.

In addition to the KITTI dataset, we use two other scene labeling datasets: Stanford Background [6] and SIFT Flow [15]. The Stanford Background dataset contains 715 images of outdoor scenes having 9 classes. Each image has a resolution of 320×240 pixels. We randomly split the images to keep 80 % of them for training and 20 % for testing. From these images, we extract a total of 40 millions patches. The SIFT Flow dataset contains 2688 manually labeled images of 256×256. The dataset has 2488 training and 200 test images containing 33 classes of objects. From this we extract 160 millions patches.

4.1 Segmentation Results with Different Training Strategies

Table 1 and Fig. 3 show the results obtained for all our training strategies. We report *global accuracy*, *average accuracy* and the *Intersection over Union* (IoU) measure. *Global* is the number of correctly classified pixels (*True Positive*) over the total number of pixels (also called *recall* or *pixel accuracy*), *Average* is the average of this recall per class (also called the *class accuracy*) and *IoU* is the ratio $TP/(TP + FP + FN)$ where TP is the number of *True Positive* and FP, FN are respectively the *False Positive* and *False Negative* averaged across all classes All results (global, average and IoU) are averaged over the considered datasets, taking into account their relative number of patches.

Table 1. Pixel (*Global*) and class (*Avg.*) accuracies and Intersection over Union (IoU) results for the combinations of each pair of datasets: 7 KITTI (7 heterogeneously labeled subsets of the KITTI dataset), Stanford and Siftflow with different training strategies: NF=No Fusion (see Fig. 1a); JT= Joint training (see Fig. 1b); DE=Dataset Equilibrium; GR=Gradient Reversal. Best results are highlighted in bold.

	7 KITTI			Stanford + SiftFlow			7 KITTI + Stanford			7 KITTI + SiftFlow		
Methods	Global	Avg	IoU	Global	Avg	IoU	Global	Avg	IoU	Global	Avg	IoU
No Fusion	77.35	54.51	41.99	72.66	33.24	25.84	77.09	55.64	43.10	76.72	45.18	34.73
JT	80.71	58.62	46.21	73.99	36.52	28.31	79.64	58.97	46.43	79.1	48.08	37.52
JT + GR	-	-	-	73.26	34.07	26.37	79.82	58.82	46.43	79.65	48.08	**37.76**
JT + DE	**80.84**	**58.89**	**46.32**	73.51	**37.08**	**28.71**	80.02	**59.23**	46.78	78.37	45.51	35.84
JT + GR + DE	-	-	-	**74.17**	35.89	27.60	**80.46**	59.02	**46.91**	**79.65**	**48.33**	37.62

Fig. 3. (a): ground truth image from [18]. **(b)**, **(c)**: our pixel classification results for the corresponding image using the JT+DE strategy (see Table 1) for the label-set outputs corresponding respectively to [18] and to [13].

No Fusion. The first learning strategy implemented consists in learning one network per dataset with the architecture described in Sect. 4 and illustrated in Fig. 1a. This is our baseline, and the results for this strategy are shown in the rows described as *No Fusion*. State of the art performances for the different KITTI sub datasets are (respectively for global and average accuracies): (92.77, 68.65) for He et al. [9]; (97.20, non reported) for Kundu et al. [13]; (82.4, 72.2) for Ladicky et. al. [14]; (51.2, 61.6) for Ros et al. [18]; (90.9, 92.10) for Sengupta et al. [19]; (non reported, 61.6) for Xu et al. [22]; and (89.3, 65.4) for Zhang et al. [25]. For Stanford with 8 classes (resp. SIFT Flow), [20] reports a global accuracy of 82.3 (resp. 80.9), a pixel accuracy of 79.1 (resp. 39.1) and an IoU of 64.5 (resp. 30.8). These isolated results are better (except for Ros et al. in Table 1) than the averages reported in our tables. This can be explained by the fact that: [13,14,18–20] only show results computed from a subset of their labels (e.g. the label *pedestrian* is ignored in [14,18,19]); the features used by all methods on KITTI are richer (e.g. depth and time); and the proposed methods always combine multiple classifiers tuned on one particular sub-dataset. To assess our contributions, we believe that *No Fusion* is the fairest baseline.

Joint Training (JT). The second alternative strategy consists in learning one single network (illustrated in Fig. 1b) with all the datasets using the selective loss function detailed in Sect. 3. We can see that this strategy gives better results than our baseline for all combination of datasets (for example, in Table 1, learning with all the subsets of KITTI gives, on average for all the 7 subsets, an improvement of +3.36 on the Global accuracy, of +4.11 on the Average accuracy and of +4.22 on the IoU). These results confirm that *Joint Training* allows us to increase the number of data used to train the network even if the datasets have different label-sets. For the sake of completeness and to evaluate the contribution of the selective loss over the mere augmentation of data, we trained our network with pairs of datasets where one dataset was used to initialize the weights of the convolutional part of the network and the other (usually the smaller one) was used to fine-tune the network. The results, not given here for lack of space, consistently show that this fine-tuning approach increases all the performance measures but at a lower extent than when using our method.

Gradient Reversal (GR) and Dataset Equilibrium (DE). As explained in Sect. 3, our selective loss does not full exploit datasets with different distributions. Thus we combine it with the gradient reversal techniques. Using gradient reversal for the KITTI dataset does not make sense since the sub-datasets all come from the same distribution. Table 1 shows that adding this gradient reversal does not always improve the performance (e.g., when learning with SIFT Flow and Stanford, the performance measures are worse than for *JT*). Starting from the intuition that unbalanced datasets can be an issue for *GR*, we experimented with weighting the contribution of each patch on the gradient computation depending on the size of the dataset this patch come from. The results show the importance of this *Dataset Equilibrium* step, even for the KITTI

dataset alone. Most best results are obtained when the joint training approach is combined both with Dataset Equilibrium and with Gradient Reversal.

4.2 Detailed Analysis on Correlations Across Tasks

We computed a label correlation matrix for all sub-datasets of the KITTI dataset, shown in Fig. 4, 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. A diagonal is visible *inside* each block, including *off-diagonal blocks*, in particular for the first 5 labels of each block. This means that, as expected, these first 5 labels are all correlated across datasets. For instance, the label *Road* from He et al. is correlated with the label *Road* from Kundu et al. with

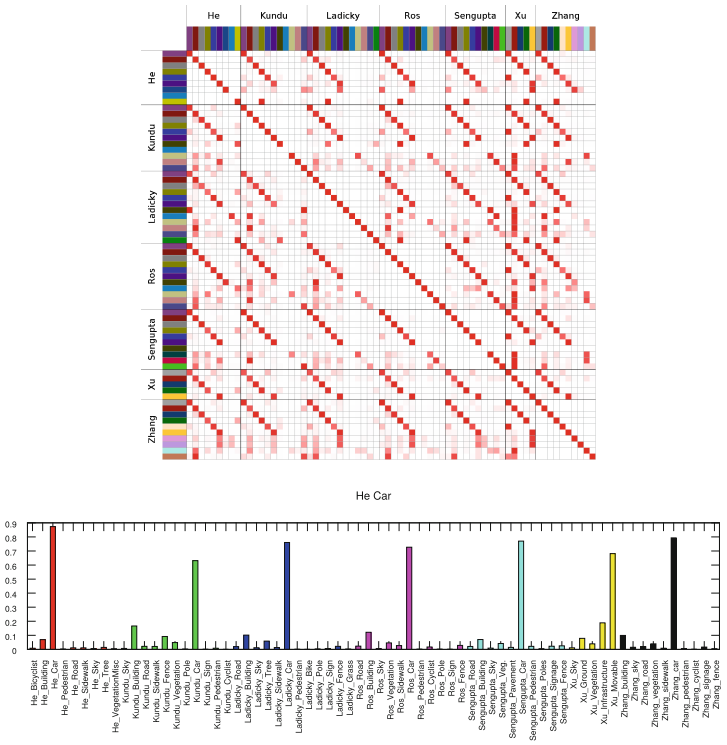


Fig. 4. Left: the empirical label correlation matrix. Each line corresponds to the average of the predictions of the network for a given target class (among the 68 labels given in Fig. 2). Darker cells indicate higher probabilities. Non-diagonal red cells correspond to labels highly correlated with the target (main diagonal) label. Below: detailed line of the correlation matrix corresponding to the label Car from the sub-dataset of He. (Color figure online)

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 Sengupta et al. have (as already discussed in Sect. 4) 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 label 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*.

5 Conclusion

In this paper, we considered the problem of multi-task multi-domain learning: we want to exploit multiple datasets that have related inputs (e.g. images) but that have been annotated for different tasks. This problem is important in two major situations: to fuse existing (small) datasets and to reuse existing dataset(s) for a related custom (new) task. We introduced a new *selective loss* function for deep networks that makes it possible to learn jointly across different tasks. We provide experimental results on semantic segmentation computer vision tasks with a total of 9 datasets. The results show that our approach allows to jointly learn from multiple datasets and to outperform per-task learning and classical fine-tuning based approaches. We also show that the domain adaptation methods (gradient reversal) can be applied for multi-task multi-domain learning but needs to be used with care and requires to balance the different datasets. An important perspective of this work is to design a strategy to practically take into account the correlations between labels highlighted by our method.

Acknowledgment. Authors acknowledge the support from the ANR project SoLStiCe (ANR-13-BS02-0002-01). They also want to thank Nvidia for providing two Titan X GPU.

References

1. Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. Mach. Learn. **79**(1–2), 151–175 (2010)
2. Farabet, C., Couprie, C., Najman, L., LeCun, Y.: Learning hierarchical features for scene labeling. IEEE TPAMI **35**(8), 1915–1929 (2013)
3. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: ICML (2015)
4. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: the KITTI dataset. Int. J. Robot. Res. **34**, 727–743 (2013)
5. Germain, P., Habrard, A., Laviolette, F., Morvant, E.: A new PAC-Bayesian perspective on domain adaptation. In: Proceedings of the 33rd International Conference on Machine Learning (ICML 2016), New York, NY, United States (2016). https://hal.archives-ouvertes.fr/hal-01307045

6. Gould, S., Fulton, R., Koller, D.: Decomposing a scene into geometric and semantically consistent regions. In: ICCV (2009)
7. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: NIPS Deep Learning Workshop (2014)
8. Hinton, G., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Improving neural networks by preventing co-adaptation of feature detectors arXiv:1207.0580 (2012)
9. Hu, H., Ben, U.: Nonparametric semantic segmentation for 3D street scenes. In: Intelligent Robots and Systems (IROS) (2013)
10. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: ICML (2015)
11. Kekec, T., Emonet, R., Fromont, E., Trémeau, A., Wolf, C.: Contextually constrained deep networks for scene labeling. In: BMVC (2014)
12. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012)
13. Kundu, A., Li, Y., Dellaert, F., Li, F., Rehg, J.M.: Joint semantic segmentation and 3D reconstruction from monocular video. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8694, pp. 703–718. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10599-4_45
14. Ladicky, L., Shi, J., Pollefeys, M.: Pulling things out of perspective. In: CVPR (2014)
15. Liu, C., Yuen, J., Torralba, A.: Nonparametric scene parsing via label transfer. IEEE TPAMI **33**(5), 978–994 (2011)
16. Mansour, Y., Mohri, M., Rostamizadeh, A.: Domain adaptation: learning bounds and algorithms. In: COLT 2009 - The 22nd Conference on Learning Theory, Montreal, Quebec, Canada, 18–21 June 2009
17. Pinheiro, P., Collobert, R.: Recurrent convolutional neural networks for scene labeling. In: ICML (2014)
18. Ros, G., Ramos, S., Granados, M., Bakhtiary, A., Vazquez, D., Lopez, A.M.: Vision-based offline-online perception paradigm for autonomous driving. In: Winter Conference on Applications of Computer Vision (WACV) (2015)
19. Sengupta, S., Greveson, E., Shahrokni, A., Torr, P.H.: Urban 3D semantic modelling using stereo vision. In: ICRA (2013)
20. Sharma, A., Tuzel, O., Jacobs, D.W.: Deep hierarchical parsing for semantic segmentation. In: CVPR (2015)
21. Tzeng, E., Hoffman, J., Darrell, T., Saenko, K.: Simultaneous deep transfer across domains and tasks. In: ICCV (2015)
22. Xu, P., Davoine, F., Bordes, J.B., Zhao, H., Denoeux, T.: Information fusion on oversegmented images: an application for urban scene understanding. In: IAPR MVA (2013)
23. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: NIPS (2014)
24. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 818–833. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10590-1_53
25. Zhang, R., Candra, S.A., Vetter, K., Zakhor, A.: Sensor fusion for semantic segmentation of urban scenes. In: IEEE ICRA (2015)
26. Zhang, X., Yu, F.X., Chang, S., Wang, S.: Deep transfer network: unsupervised domain adaptation. arXiv (2015)