Joint Graph Decomposition & Node Labeling: Problem, Algorithms, Applications – Supplement –

Evgeny Levinkov[1], Jonas Uhrig[3,4], Siyu Tang[1,2], Mohamed Omran[1], Eldar Insafutdinov[1], Alexander Kirillov[5], Carsten Rother[5], Thomas Brox[4], Bernt Schiele[1] and Bjoern Andres[1]

[1]Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany
[2]Max Planck Institute for Intelligent Systems, Tübingen, Germany [3]Daimler AG R&D, Sindelfingen, Germany
[4]Computer Vision Lab, University of Freiburg, Germany [5]Computer Vision Lab, Technische Universität Dresden, Germany

1. Implementation Details

Func. 1 and 3 choose local transformations that decrease the cost optimally. Our implementation computes cost differences incrementally, as proposed by Kernighan and Lin [8]. The exact computations are described below.

Transforming the Labeling. Func. 1 repeatedly chooses a node $\hat{v}$ and a label $\hat{l}$ such that labeling $\hat{v}$ with $\hat{l}$ decreases the cost maximally. I.e., Func. 1 repeatedly solves the optimization problem

$$(\hat{v}, \hat{l}) \in \operatorname*{argmin}_{(v,l) \in V \times L} \varphi(x^{T_{vl}(\lambda_t)}, y^{\mu_t}) - \varphi(x^{\lambda_t}, y^{\mu_t}) \quad . \tag{1}$$

While $\varphi(x^{\lambda_t}, y^{\mu_t})$ is constant, it is more efficient to minimize the difference $\varphi(x^{T_{vl}(\lambda_t)}, y^{\mu_t}) - \varphi(x^{\lambda_t}, y^{\mu_t})$ than to minimize $\varphi(x^{T_{vl}(\lambda_t)}, y^{\mu_t})$, as the difference can be computed locally, considering only the neighbors w of v in G':

$$\begin{aligned}
&\varphi(x^{T_{vl}(\lambda_t)}, y^{\mu_t}) - \varphi(x^{\lambda_t}, y^{\mu_t}) \\
&= c_{vl} - c_{v\lambda_t(v)} \\
&\quad + \sum_{vw \in A} (1 - y_{\{v,w\}}) \left(c^{\sim}_{vw,l\lambda_t(w)} - c^{\sim}_{vw,\lambda_t(v)\lambda_t(w)} \right) \\
&\quad + \sum_{wv \in A} (1 - y_{\{v,w\}}) \left(c^{\sim}_{wv,\lambda_t(w)l} - c^{\sim}_{wv,\lambda_t(w)\lambda_t(v)} \right) \\
&\quad + \sum_{vw \in A} y_{\{v,w\}} \left(c^{\nsim}_{vw,l\lambda_t(w)} - c^{\nsim}_{vw,\lambda_t(v)\lambda_t(w)} \right) \\
&\quad + \sum_{wv \in A} y_{\{v,w\}} \left(c^{\nsim}_{wv,\lambda_t(w)l} - c^{\nsim}_{wv,\lambda_t(w)\lambda_t(v)} \right) \\
&=: \Delta_{t,vl} \quad .
\end{aligned} \tag{2}$$

Initially, i.e., for $t = 0$, we compute $\Delta_{0,vl}$ for every node v and every label l. In subsequent iterations, i.e., for $t \in \mathbb{N}$ and the minimizer $(\hat{v}, \hat{l})$ of (1) chosen in this iteration, we update cost differences for all neighbors w of $\hat{v}$ in G' and all labels $l \in L$. The update rule is written below for an edge $(w, \hat{v}) \in A$. The update for an edge in the opposite direction is analogous. Below, (3) subtracts the costs due to $\hat{v}$ being labeled $\lambda_t(\hat{v})$ (which is possibly outdated), while (4) adds the costs due to $\hat{v}$ having obtained a new and possibly different label $\hat{l}$.

$$\begin{aligned}
\Delta_{t+1,wl} =& \Delta_{t,wl} \\
&- (1 - y_{\{w,\hat{v}\}}) \left(c^{\sim}_{w\hat{v},l\lambda_t(\hat{v})} - c^{\sim}_{w\hat{v},\lambda_t(w)\lambda_t(\hat{v})} \right) \\
&- y_{\{w,\hat{v}\}} \left(c^{\nsim}_{w\hat{v},l\lambda_t(\hat{v})} - c^{\nsim}_{w\hat{v},\lambda_t(w)\lambda_t(\hat{v})} \right) \qquad (3) \\
&+ (1 - y_{\{w,\hat{v}\}}) \left(c^{\sim}_{w\hat{v},l\hat{l}} - c^{\sim}_{w\hat{v},\lambda_t(w)\hat{l}} \right) \\
&+ y_{\{w,\hat{v}\}} \left(c^{\nsim}_{w\hat{v},l\hat{l}} - c^{\nsim}_{w\hat{v},\lambda_t(w)\hat{l}} \right) \quad . \qquad (4)
\end{aligned}$$

We solve (1) by means of a priority queue in time complexity $\mathcal{O}(|V| \log |V| + |V|(|L| + \log |V|) \deg G')$ with $\deg G'$ the node degree of G'. For sparse graphs and constant number $|L|$ of labels, this is $\mathcal{O}(|V| \log |V|)$.

Transformation of Labeling and Decomposition. The algorithm KLj of [9] for the minimum cost lifted multicut problem generalizes the Kernighan-Lin-Algorithm [8] for the minimum cost multicut problem. The algorithms KLj/r and KLj∗r we define further generalize KLj to the NL-LMP. The critical part is Func. 3 that solves the optimization problem

$$(\hat{v}, \hat{l}) \in \operatorname*{argmax}_{(v,l) \in V_t \times L} \varphi(x^{T_{vl}(\lambda_t)}, y^{T'_{vm'}(\mu_t)}) - \varphi(x^{\lambda}_t, y^{\mu_t}) \tag{5}$$

Let us consider w.l.o.g. two sets of vertices A and B representing two neighboring components of the graph G. Then we compute $\forall v \in A \cup B, \forall l \in L$:

$$\Delta_{vl} = c_{v\lambda_t(v)} - c_{vl} + \tag{6}$$
$$\begin{cases} \sum_{w \in A \setminus \{v\}} & c^{\sim}_{vw,\lambda_t(v)\lambda_t(w)} - c^{\not\sim}_{vw,l\lambda_t(w)} \\ \sum_{w \in B} & c^{\not\sim}_{vw,\lambda_t(v)\lambda_t(w)} - c^{\sim}_{vw,l\lambda_t(w)} \\ \sum_{w \notin A \cup B} & c^{\not\sim}_{vw,\lambda_t(v)\lambda_t(w)} - c^{\not\sim}_{vw,l\lambda_t(w)} \end{cases} , \tag{7}$$

where $w \in \mathcal{N}_{G'}(v)$. In eq. (7) the first two cases are exactly the same as given in [8] for the edges *between* partitions A and B. But in our case changing vertex's class label may affect the cut costs of edges between A and B and any other partition. Also, we have join and cut costs.

Let us assume w.l.o.g. that vertex $\hat{v}$ was chosen to be moved from set A to set B, i.e. $A = A \setminus \{\hat{v}\}$. Now we can update the expected gains of $\forall w \in \mathcal{N}_{G'}(\hat{v}), \forall l \in L$:

$$\begin{aligned} \Delta_{wl} = \Delta_{wl} &- \left(c^{\sim}_{\hat{v}w,\lambda_t(\hat{v})\lambda_t(w)} - c^{\not\sim}_{\hat{v}w,\lambda_t(\hat{v})l} \right) \\ &+ c^{\not\sim}_{\hat{v}w,\hat{l}\lambda_t(w)} - c^{\sim}_{\hat{v}w,\hat{l}l} \quad , \quad \text{if } w \in A \ , \end{aligned} \tag{8}$$

$$\begin{aligned} \Delta_{wl} = \Delta_{wl} &- \left(c^{\not\sim}_{\hat{v}w,\lambda_t(\hat{v})\lambda_t(w)} - c^{\sim}_{\hat{v}w,\lambda_t(\hat{v})l} \right) \\ &+ c^{\sim}_{\hat{v}w,\hat{l}\lambda_t(w)} - c^{\not\sim}_{\hat{v}w,\hat{l}l} \quad , \quad \text{if } w \in B \ . \end{aligned} \tag{9}$$

After that $B = B \cup \{\hat{v}\}$. In the above equations, the expression in parenthesis cancels the current contribution for vertex w, that assumed $\hat{v}$ was labeled $\lambda_t(v)$ and belonged to partition A. For the case when $|L| = 1$ and $c^{\not\sim} = c^{\sim}$ the above equations simplify to exactly the ones as in [8], but multiplied by 2, because in our objective we have two terms that operate on the edges simultaneously.

As we generalize [9] by an additional loop over the set L of labels, the analysis of the time complexity carries over from [9] with an additional multiplicative factor $|L|$.

2. Articulated Human Body Pose Estimation

2.1. Problem Statement

Pishchulin et al. [12] introduce a binary cubic problem w.r.t. a set C of body joint classes and a set D of putative detections of body joints. Every feasible solution is a pair (x, y) with $x : D \times C \to \{0, 1\}$ and $y : \binom{D}{2} \to \{0, 1\}$, constrained by the following system of linear inequalities:

$$\forall d \in D \forall cc' \in \tbinom{C}{2} : \quad x_{dc} + x_{dc'} \leq 1 \tag{10}$$

$$\begin{aligned} \forall dd' \in \tbinom{D}{2} : \quad & y_{dd'} \leq \sum_{c \in C} x_{dc} \\ & y_{dd'} \leq \sum_{c \in C} x_{d'c} \end{aligned} \tag{11}$$

$$\forall dd'd'' \in \tbinom{D}{3} : \quad y_{dd'} + y_{d'd''} - 1 \leq y_{dd''} \tag{12}$$

$\lvert V\rvert$	Alg.	Head	Sho	Elb	Wri	Hip	Knee	Ank	AP
150	[7]	84.9	79.2	66.4	52.3	65.5	59.2	51.2	65.5
	KLj/r	**87.1**	80.0	**66.8**	**53.6**	66.1	60.0	51.8	66.5
	KLj∗r	86.8	**80.2**	67.5	53.5	**66.3**	**60.3**	**51.9**	**66.6**
420	KLj/r	**90.2**	**85.2**	71.5	59.5	**71.3**	**63.1**	53.1	**70.6**
	KLj∗r	89.8	**85.2**	**71.8**	**59.6**	71.1	63.0	**53.5**	**70.6**

Table 1: Comparison of B&C [7], KLj/r and KLj∗r in an application to the task of articulated human body pose estimation.

The objective function has the form below with coefficients α and β.

$$\sum_{d \in D} \sum_{c \in C} \alpha_{dc}\, x_{dc} + \sum_{dd' \in \binom{D}{2}} \sum_{c,c' \in C} \beta_{dd'cc'} x_{dc} x_{d'c'} y_{dd'} \tag{13}$$

We identify the solutions of this problem with the solutions of the NL-LMP w.r.t. the complete graphs $G = G' = (D, \binom{D}{2})$, the label set $L = C \cup \{\epsilon\}$ and the costs $c^{\not\sim} = 0$ and

$$c_{vl} := \begin{cases} \alpha_{vl} & \text{if } l \in C \\ 0 & \text{if } l = \epsilon \end{cases} \tag{14}$$

$$c^{\sim}_{vw,ll'} := \begin{cases} \beta_{vwll'} & \text{if } l \in C \wedge l' \in C \\ 0 & \text{if } l = \epsilon \text{ xor } l' = \epsilon \\ \infty & \text{if } l = l' = \epsilon \end{cases} . \tag{15}$$

Note that in [12], $y_{dd'} = 1$ indicates a join. In our NL-LMP, $y_{dd'} = 1$ indicates a cut.

2.2. Further Results

Quantitative results for each body joint are shown in Tab. 1. Qualitative results for the MPII Human Pose dataset are shown in Fig. 1.

3. Instance-Separating Semantic Segmentation

We tackle the problem of instance-separating semantic segmentation by adapting the approach of Uhrig et al. [15]. They propose three complementary representations, which are learned jointly by a fully convolutional network (FCN) [11], that facilitate the problem of separating individual objects: Semantics, depth, and directions towards object centers. To extract object instances, a template matching approach was proposed, followed by a proposal fusion.

Instead of template matching and clustering, we rely on a generic graphical formulation of the problem using only the three predicted output scores from the network of Uhrig et al. [15], together with a suitable formulation of unary c

Figure 1: Pose estimation results on the MPII Human Pose dataset.

and pairwise terms $c^{\sim}$ and $c^{\nsim}$. As there might be up to two million nodes for a direct mapping of pixel scores to the graph, we report performance on different down-sampled versions to reduce overall computation time and reduce the impact of noise in high resolutions. Results on KITTI were achieved on half of the input resolution, for Cityscapes we down-sample the FCN scores by a factor of eight before the graph optimization.

3.1. Cut Costs Details

To define cut costs between connected pixels in the graph, we use an equally weighted sum of the three following components:

The probability of fusing two pixels v and w of different **semantic classes** is $1 - p(\lambda(v) = a, \lambda(w) = b)$, the probability of confusing label class a and b, which was computed from the training set.

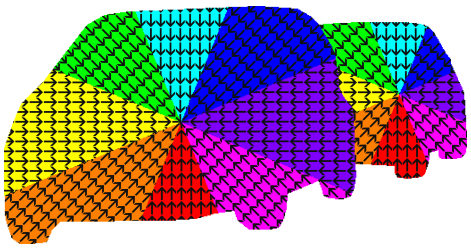

Figure 2: Instance center directions with color coding from [15]. Near object centers, directions point towards each other (center consistency). Within colored regions, directions have similar angles (angular consistency). Along object borders, directions are inconsistent in both ways.

To incorporate the depth and center direction channels, we neither use scores nor argmax predictions directly. Instead, we weight the predicted softmax scores for all non-background classes with their corresponding class to recover a continuous center direction and depth map. As objects at different **depth values** should be separated, we generate higher cut probabilities for those pixels. From training data, we found the probability of splitting two neighboring pixels to be one when the predicted depth values differ by more than 1.6 units.

If **center directions** have opposite orientations, there should be a high probability for splitting the two pixels. However, opposite directions also appear at the center of an object. Therefore, we define the cut probability as the minimum of an angular inconsistency, which punishes two pixels that point at different directions, as well as a center inconsistency, which punishes if two pixels do not point at each other, *c.f.* Fig. 2. This induces high cut probabilities at the borders of objects, as directions of pixels should have opposite center direction. The probability of splitting two neighbors due to direction inconsistency was found to be one at 90 degrees.

3.2. Dataset Specifics

For the KITTI dataset [2, 5], the only pixel-level annotated object class is *car*. For Cityscapes [4] however, there are 8 different object classes (*person, rider, car, truck, bus, train, motorcycle, bicycle*), together with 11 background classes versus 1 background class for KITTI. We found that the network model used by Uhrig et al. [15] performs close to optimum for semantic labeling on KITTI data, however has some flaws on Cityscapes.

Therefore we chose a more sophisticated network structure, which performs much better on the many different classes on Cityscapes. We use a ResNet [6] with dilated convolutions [3] for cut costs c, namely the unary terms consisting of scores for the problem of semantic labeling, which was trained independently on the Cityscapes dataset [4].

To obtain the unaries for Cityscapes, we use a slightly modified ResNet-50 network. We introduce dilated convolutions in the conv4_x and conv5_x layers to increase the output resolution from $1/32$ to $1/8$ of the input resolution. We then remove the final average pooling layer and for classification use a convolutional layer with 5×5 dilated kernels with a dilation size of 12. This is identical to the best performing basic ResNet-50 variant reported in ([16], Table 1).

Due to GPU memory constraints, we train with $512px \times 768px$ crops randomly sampled from the full-resolution training set images. We apply minimal data augmentation, i.e. random horizontal flips, and train with a batch size of 5. We train the network for 60000 iterations using the Adam solver with an initial learning rate of 0.000025, weight decay of 0.0005 and momentum of 0.9. We use the "poly" learning rate policy to gradually reduce the learning rate during training with the power parameter set to 0.9, which as reported in both [10] and [1] yields better results than the commonly used "step" reduction policy.

At test-time we apply the network to overlapping $1024px \times 768px$ crops of the full-resolution test set images and stitch the results to obtain the final predictions.

For KITTI however, we stick with the original semantic scores. The only adaptation for our definition of the semantic cut costs c is an additional weighting of the semantic scores: As depth and center directions are only estimated for objects, all three channels contain knowledge of the objectness of a certain pixel. We therefore use the semantic scores weighted by the depth and direction scores for objects as unaries. This increases robustness of the semantics as all three channels must agree to achieve high scores.

3.3. Post Processing

Using the unary and pairwise terms defined above, we solve the graph for labels and components with our proposed algorithms KLj/r and KLj$*$r. As the center direction representation inherently cannot handle cases of full occlusions, e.g. if a bicycle is split into two connected components by a pedestrian in front of it, we apply a similar component fusion technique as proposed in [15]: We accumulate direction predictions within each component and fuse it with another suitable component when direction predictions are clearly overshooting into a certain direction. We compare performance of the raw graph output as well as the fused instances in Tab. 2 (top).

3.4. Detailed Results

As there are different metrics used by related approaches, we report performance on the Cityscapes [4] and KITTI [2, 5] dataset using both proposed metrics. The instance score required for the evaluation on Cityscapes was chosen as the size of the instance in pixels multiplied by its mean depth - this score achieved slightly better results compared to a

Algorithm	Dataset	AP	$AP^{50\%}$
Ours KLj/r (raw)	KITTI val	43.0	72.5
Ours KLj∗r (raw)	KITTI val	43.5	72.6
Ours KLj/r (fused)	KITTI val	**50.5**	**82.9**
Ours KLj∗r (fused)	KITTI val	50.3	82.4
Pixel Encoding [15]	KITTI test	41.6	69.1
Ours KLj∗r (fused)	KITTI test	**43.6**	**71.4**

Table 2: Comparison of algorithms for instance segmentation on the KITTI [2] datasets using the mean average precision metrics introduced in [4].

Alg.	IoU	AvgFP	AvgFN	InsPr	InsRe	InsF1
[18]	77.4	0.479	0.840	48.9	43.8	46.2
[17]	77.0	0.375	1.139	65.3	50.0	56.6
[15]	84.1	0.201	0.159	**86.3**	74.1	**79.7**
[13]	**87.4**	**0.118**	0.278	-	-	-
Ours	83.9	0.555	**0.111**	69.2	**76.5**	72.7

Table 3: Comparison of algorithms for instance segmentation on the KITTI test dataset [2] using metrics proposed in [17]. Ours describes the performance of our KLj∗r variant.

constant score.

For KITTI, we outperform all existing approaches using the Cityscapes metric (without adapting the semantic scores of Uhrig et al. [15]), which averages precision and recall performance for multiple overlaps, *c.f*. Tab. 2 (bottom). We evaluate the performance using KLj/r or KLj∗r and raw graph output (raw) or the post-processed results using above described fusion (fused) in Tab. 2 (top). Using the KITTI metrics, we perform among the best results while having a slight preference of Recall over Precision, *c.f*. Tab. 3.

For Cityscapes, we report evaluation metrics using both the raw scores of Uhrig et al. [15] as well as our final proposed model using the semantic scores of a ResNet [6] together with the center direction and depth scores of Uhrig et al. [15], *c.f*. Tab. 5 (top). Using our adapted ResNet version, we outperform the currently published state-of-the art, *c.f*. Tab. 5 (bottom). Note that we report significantly better performance for the large vehicle classes truck, bus, and trains despite starting from the same FCN output, *c.f*. Tab. 4. This comes from incorporating confusion probabilities between unreliable classes as well as optimizing jointly for semantics and instances.

3.5. Qualitative Results

See Fig. 3 for some qualitative results for our instance-separating semantic segmentation on the Cityscapes validation dataset [4]. It can be seen that we perform equally well for large and small objects, we only tend to fuse pedestrians too often, which explains the worse performance on pedestrians - *c.f*. the mother with her child on the right in the last row of Fig. 3. Also, the impact of the proposed post-processing based on the fusion algorithm proposed by Uhrig et al. [15] can be seen clearly: Due to noisy predictions, the raw graph output is often highly over-segmented. However, after applying the fusion step, most objects are correctly fused.

	person	rider	car	truck	bus	train	motorcycle	bicycle
[4]	5.6	3.9	26.0	13.8	**26.3**	15.8	8.6	3.1
[15]	**31.8**	**33.8**	37.8	7.6	12.0	8.5	20.5	**17.2**
Ours	18.4	29.5	**38.3**	**16.1**	21.5	**24.5**	**21.4**	16.0

Table 4: Comparison of performance on Cityscapes test using the mean average precision metric $AP^{50\%}$ [4]. Ours describes the performance of our KLj∗r (ResNet) variant.

Algorithm	Dataset	AP	$AP^{50\%}$
Pixel Encoding [15]	CS val	9.9	22.5
Ours ([15] scores)	CS val	9.4	22.1
Ours KLj/r (ResNet)	CS val	11.3	**26.8**
Ours KLj∗r (ResNet)	CS val	**11.4**	26.1
MCG+R-CNN [4]	CS test	4.6	12.9
Pixel Encoding [15]	CS test	8.9	21.1
Ours KLj∗r (ResNet)	CS test	**9.8**	**23.2**

Table 5: Comparison of algorithms for instance segmentation on the Cityscapes (CS) dataset [4] using the mean average precision metrics introduced in [4].

3.6. Outlook

The reason for the varying performance for objects of different semantic classes certainly comes from their very different typical forms, which we do not incorporate in our general approach. Uhrig et al. [15] use different aspect ratios for their sliding object templates to cope for these changes. In future work, we would like to combine multiple graphs for different semantic classes to boost individual class performance. Also, the predicted FCN representation and scores will be adjusted for better suiting the requirements of our graph optimization.

4. Multiple Object Tracking

4.1. Problem Statement

Tang et al. [14] introduce a binary linear program w.r.t. a graph $G = (V, E)$ whose nodes are candidate detections of humans visible in an image. Every feasible solution is a pair (x, y) with $x \in \{0, 1\}^V$ and $y \in \{0, 1\}^E$, constrained such

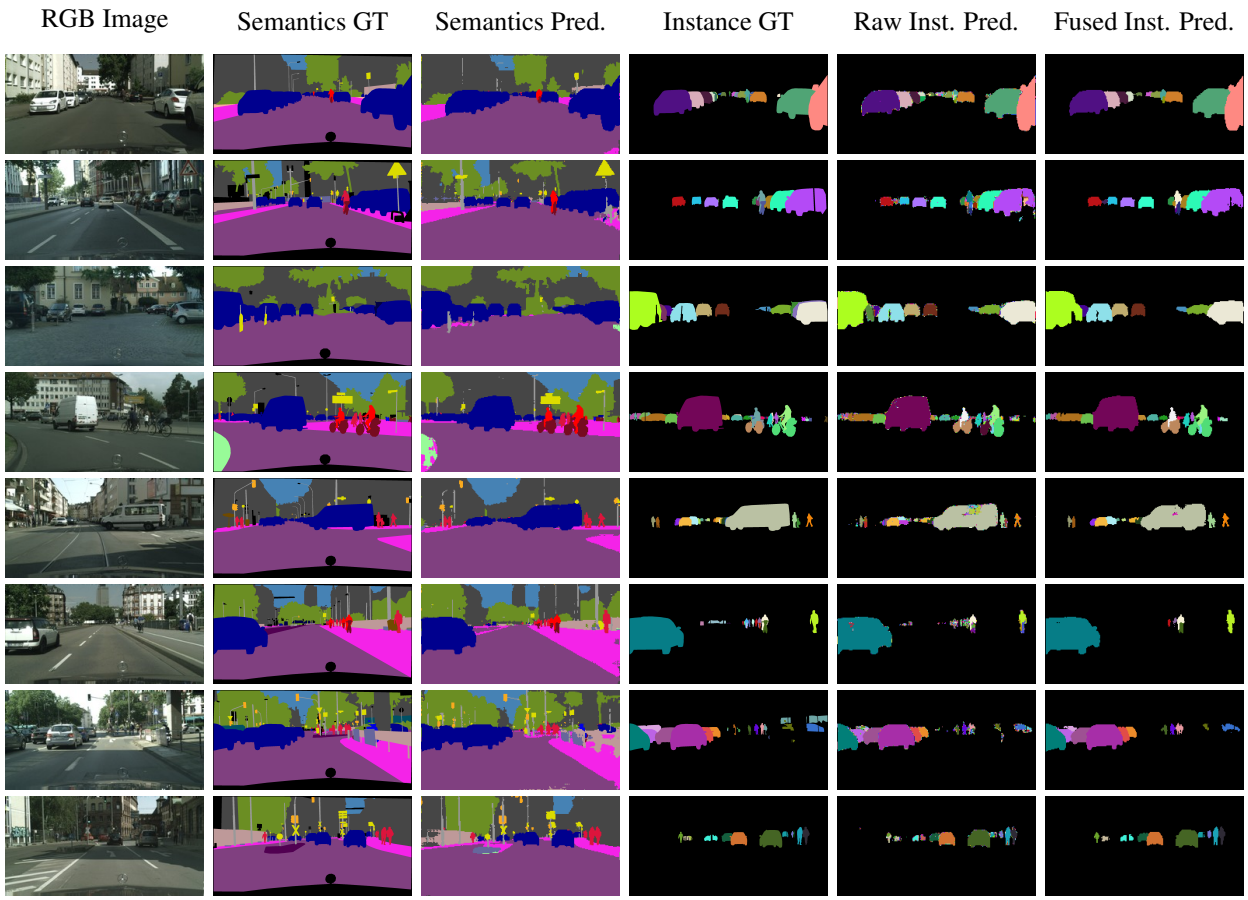

Figure 3: Visualization of our predictions on the Cityscapes validation dataset [4], where we can compare with corresponding ground truth (GT) and show respective RGB images.

that

$$\forall \{v,w\} \in E: \quad y_{vw} \leq x_v \tag{16}$$

$$y_{vw} \leq x_w \tag{17}$$

$$\forall C \in \text{cycles}(G)\ \forall e \in C: \quad 1 - y_e \leq \sum_{f \in C \setminus \{e\}} (1 - y_f) \tag{18}$$

The objective function has the form below with coefficients α and β.

$$\sum_{v \in V} \alpha_v x_v + \sum_{e \in E} \beta_e y_e \tag{19}$$

We identify the solutions of this problem with the solutions of the NL-LMP w.r.t. the graphs $G' = G$, the label set $L = \{\epsilon, 1\}$ and the costs $c^{\nearrow} = 0$ and

$$c_{vl} := \begin{cases} \alpha_v & \text{if } l = 1 \\ 0 & \text{if } l = \epsilon \end{cases} \tag{20}$$

$$c^{\sim}_{vw,ll'} := \begin{cases} \beta_{vw} & \text{if } l = 1 \wedge l' = 1 \\ 0 & \text{if } l = 1 \text{ xor } l' = 1 \\ \infty & \text{if } l = l' = \epsilon \end{cases} . \tag{21}$$

Note that in [14], $y_{dd'} = 1$ indicates a join. In our NL-LMP, $y_{dd'} = 1$ indicates a cut.

4.2. Further Results

A complete evaluation of our experimental results in terms of the Multiple Object Tracking Challenge 2016 can be found at http://motchallenge.net/tracker/NLLMPa.

References

[1] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *CoRR*, abs/1606.00915, 2016. 4

[2] L. C. Chen, S. Fidler, and R. Urtasun. Beat the MTurkers: Automatic image labeling from weak 3d supervision. In *CVPR*, 2014. 4, 5

[3] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv:1606.00915*, 2016. 4

[4] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The Cityscapes Dataset for semantic urban scene understanding. In *CVPR*, June 2016. 4, 5, 6

[5] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR*, 2012. 4

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4, 5

[7] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele. DeeperCut: A deeper, stronger, and faster multi-person pose estimation model. In *ECCV*, 2016. 2

[8] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *Bell Systems Technical Journal*, 49:291–307, 1970. 1, 2

[9] M. Keuper, E. Levinkov, N. Bonneel, G. Lavoué, T. Brox, and B. Andres. Efficient decomposition of image and mesh graphs by lifted multicuts. In *ICCV*, 2015. 1, 2

[10] W. Liu, A. Rabinovich, and A. C. Berg. Parsenet: Looking wider to see better. *CoRR*, abs/1506.04579, 2015. 4

[11] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 2

[12] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. Gehler, and B. Schiele. DeepCut: Joint subset partition and labeling for multi person pose estimation. In *CVPR*, 2016. 2

[13] M. Ren and R. Zemel. End-to-end instance segmentation and counting with recurrent attention. In *arXiv:1605.09410 [cs.CV]*, 2016. 5

[14] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Subgraph decomposition for multi-target tracking. In *CVPR*, 2015. 5, 6

[15] J. Uhrig, M. Cordts, U. Franke, and T. Brox. Pixel-level encoding and depth layering for instance-level semantic labeling. In *GCPR*, 2016. 2, 4, 5

[16] Z. Wu, C. Shen, and A. van den Hengel. Bridging category-level and instance-level semantic image segmentation. *CoRR*, abs/1605.06885, 2016. 4

[17] Z. Zhang, S. Fidler, and R. Urtasun. Instance-level segmentation with deep densely connected MRFs. In *CVPR*, 2016. 5

[18] Z. Zhang, A. G. Schwing, S. Fidler, and R. Urtasun. Monocular object instance segmentation and depth ordering with CNNs. In *ICCV*, 2015. 5