Gated Fully Fusion for Semantic Segmentation

Xiangtai Li,1 Houlong Zhao,2 Lei Han,3 Yunhai Tong,1 Shaohua Tan,1 Kuiyuan Yang2
1School of EECS, Peking University 2DeepMotion 3Tencent AI Lab
{lxtpku, yhtong}@pku.edu.cn, tan@cis.pku.edu.cn
{houlongzhao, kuiyuanyang}@deepmotion.ai
leihan.cs@gmail.com

Abstract
Semantic segmentation generates comprehensive understanding of scenes through densely predicting the category for each pixel. High-level features from Deep Convolutional Neural Networks already demonstrate their effectiveness in semantic segmentation tasks, however the coarse resolution of high-level features often leads to inferior results for small/thin objects where detailed information is important. It is natural to consider importing low level features to compensate for the lost detailed information in high-level features. Unfortunately, simply combining multi-level features suffers from the semantic gap among them. In this paper, we propose a new architecture, named Gated Fully Fusion (GFF), to selectively fuse features from multiple levels using gates in a fully connected way. Specifically, features at each level are enhanced by higher-level features with stronger semantics and lower-level features with more details, and gates are used to control the propagation of useful information which significantly reduces the noises during fusion. We achieve the state of the art results on four challenging scene parsing datasets including Cityscapes, Pascal Context, COCO-stuff and ADE20K.

Introduction
Semantic segmentation densely predicts the semantic category for every pixel in an image, such comprehensive image understanding is valuable for many vision-based applications such as medical image analysis (Ronneberger, Fischer, and Brox 2015), remote sensing (Kampffmeyer, Salberg, and Jenssen 2016) and autonomous driving (Xu et al. 2017). However, precisely predicting label for every pixel is challenging as illustrated in Fig. 1, since pixels can be from tiny or large objects, far or near objects, and inside object or object boundary.

As a semantic prediction problem, the basic task of semantic segmentation is to generate high-level representation for each pixel, i.e., a high-level and high-resolution feature map. Given the ability of ConvNets in learning high-level representation from data, semantic segmentation has made much progress by leveraging such high-level representation. However, high-level representation from ConvNets is generated along lowering the resolution, thus high-resolution and high-level feature maps are distributed in two ends in a ConvNet.

To get a feature map that is both high-resolution and high-level, which is not readily available in a ConvNet, it is natural to consider fusing high-level feature maps from top layers and high-resolution feature maps from bottom layers. These feature maps are with different properties, that high-level feature map can correctly predict most of the pixels on large patterns in a coarse manner, which is widely used in the current semantic segmentation approaches, while low-level feature maps can only predict few pixels on small patterns.

Thus, simply combining high-level feature maps and high-resolution feature maps will drown useful information in massive useless information, and cannot reach an informative high-level and high-resolution feature map. Therefore, an advanced fusion mechanism is required to collect information selectively from different feature maps. To achieve this, we propose Gated Fully Fusion (GFF) which uses gating mechanism, a kind of operation commonly used for information extraction from time series, to pixelwisely measure the usefulness of each feature vector, and control information propagation through gates accordingly. The principle of the gate at each layer is designed to either send...
out useful information to other layers or receive information from other layers when the information in the current layer is useless. Using gate to control information propagation, redundancies can also be effectively minimized in the network, allowing us to fuse multi-level feature maps in a fully-connected manner. Fig 1 compares the results of GFF and PSPNet (Zhao et al. 2017), where GFF can handle fine-level details such as poles and traffic lights in a much better way.

In addition, contextual information in large receptive field is also very important for semantic segmentation as proved by PSPNet (Zhao et al. 2017), ASPP (Chen et al. 2018a) and DenseASPP (Yang et al. 2018). Therefore, we also model contextual information after GFF to further improve the performance. Specifically, we propose a dense feature pyramid (DFP) module to encode context information into each feature map. DFP reuses the contextual information for each feature level and aims to enhance the context modeling part while GFF operates on the backbone network to capture more detailed information. Combining both components in a single end-to-end network, we achieve state-of-the-art results on four scene parsing datasets.

The main contributions of our work can be summarized as three points: Firstly, we propose Gated Fully Fusion to generate high-resolution and high-level feature map from multi-level feature maps, and Dense Feature Pyramid to enhance the semantic representation of multi-level feature maps. Secondly, detailed analysis with visualization of gates learned in different layers intuitively shows the information regulation mechanism in GFF. Finally, The proposed method is extensively verified on four standard semantic segmentation benchmarks including Cityscapes, Pascal Context, COCO-stuff and ADE20K, where our method achieves state-of-the-art performance on all four tasks. In particular, our model achieves 82.3\% mIoU on Cityscapes test set trained only on the fine labeled data with ResNet101 as backbone.

Related Work

Context modeling Though high-level feature maps in ConvNets have shown promising results on semantic segmentation (Long, Shelhamer, and Darrell 2015), their receptive field sizes are still not large enough to capture contextual information for large objects and regions. Thus, context modeling becomes a practical direction in semantic segmentation. PSPNet (Zhao et al. 2017) uses spatial pyramid pooling to aggregate multi-scale contextual information. Deeplab series (Chen et al. 2015; 2018a; 2017) develop atrous spatial pyramid pooling (ASPP) to capture multi-scale contextual information by dilated convolutional layers with different dilation rates. Instead of parallel aggregation as adopted in PSPNet and Deeplab, Yang et al. (Yang et al. 2018) and Bilinski et al. (Bilinski and Prisacariu 2018) follow the idea of the dense connection (Huang et al. 2017) to encode contextual information in a dense way. In (Peng et al. 2017), factorized large filters are directly used to increase the receptive field size for context modeling. SVCNet (Ding et al. 2019) generates a scale and shape-variant semantic mask for each pixel to confine its contextual region. In PSANet (Zhao et al. 2018), contextual information is collected from all positions according to the similarities defined in a projected feature space. Similarly, DANet (Fu et al. 2018) and CCNet (Huang et al. 2018b) use non-local style operator (Wang et al. 2018) to aggregate information from the whole image based on similarities.

Multi-level feature fusion Despite a loss of contextual information, the top layer also lacks fine detailed information. To address this issue, in FCN (Long, Shelhamer, and Darrell 2015), predictions from middle layers are used to improve segmentation for detailed structures, while hypercolumns (hariharan et al. 2015) directly combines features from multiple layers for prediction. The U-Net (Ronneberger, Fischer, and Brox 2015) adds skip connections between the encoder and decoder to reuse low level features, (Zhang et al. 2018b) improves U-Net by fusing high-level features into low-level features. Feature Pyramid Network (FPN) (Lin et al. 2017b) uses the structure of U-Net with predictions made on each level of the feature pyramid. DeepLabV3+ (Chen et al. 2018b) refines the decoder of its previous version by combing low-level features. In (Lin et al. 2018) and (Ding et al. 2018), they proposed to locally fuse every two adjacent feature maps in the feature pyramid into one feature map until only one feature map is left. These fusion methods operate locally in the feature pyramid without awareness of the usefulness of all feature maps to be fused, which limits the propagation of useful features.

Gating mechanism In deep neural networks, especially for recurrent networks, gates are commonly utilized to control the information propagation. For example, LSTM (Hochreiter and Schmidhuber 1997) and GRU (Cho et al. 2014) are two typical cases using different gates to handle long-term memory and dependencies. The highway network (Srivastava, Greff, and Schmidhuber 2015) adds skip connections between the encoder and decoder to reuse low level features from multiple layers for prediction. The U-Net (Ronneberger, Fischer, and Brox 2015) adds skip connections for scene parsing and depth estimation, PADNet (Xu et al. 2018) is proposed to use gates to fuse multilevel features trained from multiple auxiliary tasks. DepthSeg (Kong and Fowlkes 2018) proposes depth-aware gating module which uses depth estimates to adaptively modify the pooling field size in high-level feature map. Our method is related and inspired by the above methods, and differs from them in that we propose to fuse multi-level feature maps simultaneously through gating mechanism, and the resulting method surpasses the state-of-the-art approaches.

Method

In this section, we first overview the basic setting of multi-level feature fusion and three baseline fusion strategies. Then, we introduce the proposed multi-level fusion module (GFF) and the whole network with the context modeling module (DFP).

Multi-level Feature Fusion

Given L feature maps \( \{X_i \in \mathbb{R}^{H_i \times W_i \times C_i}\}_{i=1}^L \) extracted from some backbone networks such as ResNet (He et al. 2016), where feature maps are ordered by their depth in the network with increasing semantics but decreasing details,
Figure 2: The proposed gated fully fusion module, where $G_l$ is the gate map generated from $X_l$, and features corresponding high gate values are allowed to send out and regions with low gate values are allowed to receive.

$H_i$, $W_i$ and $C_i$ are the height, width and number of channels of the $i$th feature map respectively, feature maps of higher levels are with lower resolution due to the downsampling operations, i.e., $H_{i+1} \leq H_i, W_{i+1} \leq W_i$. In semantic segmentation, the top feature map $X_L$ with $1/8$ resolution of the raw input image is mostly used for its rich semantics. The major limitation of $X_L$ is its low spatial resolution without detailed information, because the outputs need to be with the same resolution as the input image. In contrast, feature maps of low level from shallow layers are with high resolution, but with limited semantics. Intuitively, combining the complementary strengths of multiple level feature maps would achieve the goal of both high resolution and rich semantics, and this process can be abstracted as a fusion process $f$, i.e.,

$$\{X_1, X_2 \cdots X_L\} \xrightarrow{f} \{\tilde{X}_1, \tilde{X}_2 \cdots \tilde{X}_L\} \quad (1)$$

where $\tilde{X}_l$ is the fused feature map for the $l$th level. To simplify the notations in following equations, bilinear sampling and $1 \times 1$ convolution are ignored which are used to reshape the feature maps at the right hand side to let the fused feature maps have the same size as those at the left hand side. Concatenation is a straightforward operation to aggregate all the information in multiple feature maps, but it mixes the useful information with large amount of non-informative features. Addition is another simple way to combine feature maps by adding features at each position, while it suffers from the similar problem as concatenation. FPN (Lin et al. 2017b) conducts the fusion process through a top-down pathway with lateral connections. The three fusion strategies can be formulated as,

- Concat: $\tilde{X}_l = \text{concat}(X_1, \ldots, X_L)$,
- Addition: $\tilde{X}_l = \sum_{i=1}^{L} X_i$,  
- FPN: $\tilde{X}_l = \tilde{X}_{l+1} + X_l$, where $\tilde{X}_L = X_L$. \quad (4)

The problem of these basic fusion strategies is that feature maps are fused together without measuring the usefulness of each feature vector, and massive useless features are mixed with useful feature during fusion.

**Gated Fully Fusion**

**GFF module design:** The basic task in multi-level feature fusion is to aggregate useful information together under interference of massive useless information. Gating is a mature mechanism to measure the usefulness of each feature vector in a feature map and aggregates information accordingly. In this paper, Gated Fully Fusion (GFF) is designed based on the simple addition-based fusion by controlling information flow with gates. Specifically, each level $l$ is associated with a gate map $G_l \in [0, 1]^{H_l \times W_l}$. With these gate maps, the addition-based fusion is formally defined as

$$\tilde{X}_l = (1 + G_l) \cdot X_l + (1 - G_l) \cdot \sum_{i=1, i \neq l}^{L} G_i \cdot X_i, \quad (5)$$

where $\cdot$ denotes element-wise multiplication broadcasting in the channel dimension, each gate map $G_l = \text{sigmoid}(w_l \ast X_l)$ is estimated by a convolutional layer parameterized with $w_l \in \mathbb{R}^{1 \times 1 \times C_l}$. There are totally $L$ gate maps where $L$ equals to the number of feature maps. The detailed operation can be seen in Fig 2.

**GFF involves duplex gating mechanism:** A feature vector at position $(x, y)$ from level $i$, (where $i \neq l$) can be fused to $l$ only when the value of $G_i(x, y)$ is large and the value of $G_l(x, y)$ is small, i.e., information is sent when level $l$ has the useful information that level $l$ is missing. Besides that useful information can be regulated to the right place through gates, useless information can also be effectively suppressed on both the sender and receiver sides, and information redundancy can be avoided because the information is only received when the current position has useless.
features. More visualization examples can be seen in experiments parts.

Comparison with Other Gate module: The work (Ding et al. 2018) also used gates for information control between adjacent layers. GFF differs in using gates to fully fuse features from every level instead of adjacent levels, and richer information in all levels with large usability variance motivates us to design the duplex gating mechanism, which filters out useless information more effectively with gates at both sides of the sender and receiver. Experimental results in the experiment section demonstrate the advantage of the proposed method.

Dense Feature Pyramid

Context modeling aims to encode more global information, and it is orthogonal to the proposed GFF because GFF is designed for backbone level. Therefore, we further design a module to encode more contextual information from outputs of both PSPNet (Zhao et al. 2017) and GFF. Motivated by that dense connections can strengthen feature propagation (Huang et al. 2017), we also densely connect the feature maps in a top-down manner starting from feature map outputted from the PSPNet, and high-level feature maps are reused multiple times to add more contextual information to low levels, which was found important in our experiments for correctly segmenting large pattern in objects. This process is shown as follows:

\[ y_i = H_i([y_0, \hat{X}_1, ..., \hat{X}_{i-1}]) \]  

Consequently, the \( j \)-th feature pyramid receives the feature-maps of all preceding pyramids, \( y_0, \hat{X}_1, ..., \hat{X}_{i-1} \) as input and outputs current pyramid \( y_i \); where \( x_0 \) is the output of PSPNet and \( \hat{X}_i \) is the output of \( i \)-th GFF module. Fusion function \( H_i \) is implemented by a single convolution layer. Since the feature pyramid is densely connected, we denote this module as Dense Feature Pyramid (DFP). The collections of DFP’s outputs \( y_i \) are used for final prediction. Both GFF and DFP can be plugged into existing FCNs for end-to-end training with only slightly extra computation cost.

Network Architecture and Implementation

Our network architecture is designed based on previous state-of-the-art network PSPNet (Zhao et al. 2017) with ResNet (He et al. 2016) as backbone for basic feature extraction, the last two stages in ResNet are modified with dilated convolution to make both strides to 1 and keep spatial information. Fig 3 shows the overall framework including both GFF and DFP. PSPNet forms the bottom-up pathway with backbone network and pyramid pooling module (PPM), where PPM is at the top to encode contextual information. Feature maps from last residual blocks in each stage of backbone are used as the input for GFF module, and all feature maps are reduced to 256 channels with \( 1 \times 1 \) convolutional layers. The output feature maps from GFF are further fused with two \( 3 \times 3 \) convolutional layers in each level before feeding into the DFP module. All convolutional layers are followed by batch normalization (Ioffe and Szegedy 2015) and ReLU activation function. After DFP, all feature maps are concatenated for final semantic segmentation. Compared with the basic PSPNet, the proposed method only slightly increases the number of parameters and computations. The entire network is trained in an end-to-end manner driving by cross-entropy loss defined on the segmentation benchmarks. To facilitate the training process, an auxiliary loss together with the main loss are used to help optimization following (Lee et al. 2015), where the main loss is defined on the final output of the network and the auxiliary loss is defined on the output feature map at stage3 of ResNet with weight of 0.4 (Zhao et al. 2017).

Experiment

In this section, we analyze the proposed method on Cityscapes (Cordts et al. 2016) dataset and report results on other datasets.

Implementation Details

Our implementation is based on PyTorch (Paszke et al. 2017). The weight decay is set to 1e-4. Standard SGD is used for optimization, and “poly” learning rate scheduling policy is used to adjust learning rate, where initial learning rate is set to 1e-3 and decayed by \( (1 - \frac{iter}{total\ iter})^{power} \) with \( power = 0.9 \). Synchronized batch normalization (Zhang et al. 2018a) is used. For Cityscapes, crop size of \( 864 \times 864 \) is used, 100K training iterations with mini-batch size of 8 is carried for training. For ADE20K, COCO-stuff and Pascal Context, crop size of \( 512 \times 512 \) is used (images with side smaller than the crop size are padded with zeros), 150K training iterations are used with mini-batch size of 16. As a common practice to avoid data augmentation including random horizontal flipping, random cropping, random color jittering within the range of \([ -10, 10 ] \), and random scaling in the range of \([ 0.75, 2 ] \) are used during training.
Experiments on Cityscapes Dataset

Cityscapes is a large-scale dataset for semantic urban scene understanding. It contains 5000 fine pixel-level annotated images, which is divided into 2975, 500, and 1525 images for training, validation and testing respectively, where labels of training and validation are publicly released and labels of testing set are held for online evaluation. It also provides 20000 coarsely annotated images. 30 classes are annotated and 19 of them are used for pixel-level semantic labeling task. Images are in high resolution with the size of 1024 × 2048. The evaluation metric for this dataset is the mean Intersection over Union (mIoU).

Strong Baseline We choose PSPNet (Zhao et al. 2017) as our baseline model which achieved state-of-the-art performance for semantic segmentation. We re-implement PSPNet on Cityscapes and achieve similar performance with mIoU of 78.6% on validation set. All results are reported by using sliding window crop prediction.

Ablation Study on Feature Fusion Methods First, we compare several methods introduced in Method part. To speed up the training process, we use weights from the trained PSPNet to initialize the parameters in each fusion method. We use train-fine data for training and report performance on validation set. For fair comparison with concatenation and addition, we also reduce the channel dimension of feature maps to 256 and use two 3 × 3 convolutional layers to refine the fused feature map. As for FPN, we implement the original FPN for semantic segmentation following (Kirillov et al. 2017) and we add it to PSPNet. Note that FPN based on PSPNet fuses 5 feature maps, where one is context feature map from pyramid pooling module and others are from the backbone.

All the results are shown in Table 1. As expected, concatenation and addition only slightly improve the baseline, and FPN achieves the best performance among the three base fusion methods, while the proposed GFF obtains even more improvement with mIoU of 80.4%. Since GFF is a gated version of addition-based fusion, the results demonstrate the effectiveness of the used gating mechanism. For further comparison, we also add the proposed gating mechanism into top-down pathway of FPN and observe slight improvement, which is reasonable since most high-level features are useful for low levels. This demonstrates the advantage of fully fusing multi-level feature maps, and the importance of gating mechanism especially during fusing low-level features to high levels. Fig 4 shows results after using GFF, where the accuracies of predictions for both distant objects and object boundaries are significantly improved.

Ablation Study for Improvement Strategies We perform two strategies to further boost the performance of our model: (1) DFP: Dense Feature Pyramid is used after the output of GFF module; and (2) MS: multi-scale inference is adopted, where the final segmentation map is averaged from segmentation probability maps with scales {0.75, 1, 1.25, 1.5, 1.75} for evaluation. Experimental results are shown in Table 2, and DFP further improves the performance by 0.8% mIoU. Fig. 5 shows several visual comparisons, where DFP generates more consistent segmentation inside large objects and demonstrates the effectiveness in using contextual information for resolving local ambiguities. With multi-scale inference, our model achieves 81.8% mIoU, which significantly outperforms previous state-of-the-art model DeepLabv3+ (79.55% on Cityscapes validation set) by 2.25%.

Ablation Study for other architectures we also perform experiments two different backbone architectures (Yang et al. 2018). One is another strong baseline and the other is PSPNet with lightweight backbone. Results are shown in Table 4. It shows that both GFF and DFP show their generality on improving model results. In particular, resnet18 based PSPNet improve 5.9% point from the baseline.

Computation Cost In Table 3, we also study the computational cost of using our modules, where our method spends 7.7% more computational cost and 6.3% more parameters compared with the baseline PSPNet.

Comparison to the State-of-the-Art As a common practice toward best performance, we average the predictions of multi-scaled images for inference. For fair comparison, all methods are only trained using fine annotated dataset.

![Figure 5: DFP enhances segmentation results on large scale objects and generates more consistent results. Best view in color and zoom in.](image)

<table>
<thead>
<tr>
<th>Method</th>
<th>mIoU(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSPNet(Baseline)</td>
<td>78.6</td>
</tr>
<tr>
<td>PSPNet + Concat</td>
<td>78.8 (0.2↑)</td>
</tr>
<tr>
<td>PSPNet + Addition</td>
<td>78.7 (0.1↑)</td>
</tr>
<tr>
<td>PSPNet + FPN</td>
<td>79.3 (0.7↑)</td>
</tr>
<tr>
<td>PSPNet + Gated FPN</td>
<td>79.4 (0.8↑)</td>
</tr>
<tr>
<td>PSPNet + GFF</td>
<td>80.4 (1.8↑)</td>
</tr>
</tbody>
</table>

Table 1: Comparison experiments on Cityscapes validation set, where PSPNet serves as the baseline method.

<table>
<thead>
<tr>
<th>Method</th>
<th>mIoU(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSPNet(Baseline)</td>
<td>78.6</td>
</tr>
<tr>
<td>PSPNet + GFF</td>
<td>80.4 (1.8↑)</td>
</tr>
<tr>
<td>PSPNet + GFF + DFP</td>
<td>81.2 (2.6↑)</td>
</tr>
<tr>
<td>PSPNet + GFF + DFP + MS</td>
<td>81.8 (3.2↑)</td>
</tr>
</tbody>
</table>

Table 2: Comparison experiments on Cityscapes validation set, where PSPNet serves as the baseline method.
Table 3: Computational cost comparison, where PSPNet serves as the baseline with image of size 512 × 512 as input.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>mIoU(%)</th>
<th>FLOPS(G)</th>
<th>Params(M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSPNet (Baseline)</td>
<td></td>
<td>78.6</td>
<td>580.1</td>
<td>65.6</td>
</tr>
<tr>
<td>PSPNet + GFF</td>
<td></td>
<td>80.4</td>
<td>600.1</td>
<td>69.7</td>
</tr>
<tr>
<td>PSPNet + GFF + DFP</td>
<td></td>
<td>81.2</td>
<td>625.5</td>
<td>70.5</td>
</tr>
</tbody>
</table>

Table 4: Ablation study on two different models, where mIoU is evaluated on Cityscapes validation set.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>mIoU(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSP</td>
<td>ResNet18</td>
<td>73.0</td>
</tr>
<tr>
<td>PSP + GFF</td>
<td>ResNet18</td>
<td>76.6</td>
</tr>
<tr>
<td>PSP + GFF + DFP</td>
<td>ResNet18</td>
<td>78.9</td>
</tr>
</tbody>
</table>

Table 5: State-of-the-art comparison experiments on Cityscapes test set. † means only using the train-fine dataset. ‡ means both the train-fine and val-fine data are used.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>mIoU(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSPNet (Zhao et al. 2017)†</td>
<td>ResNet101</td>
<td>78.7</td>
</tr>
<tr>
<td>PSPNet (Zhao et al. 2018)†</td>
<td>ResNet101</td>
<td>78.6</td>
</tr>
<tr>
<td>GFFNet (Ours)‡</td>
<td>ResNet101</td>
<td>80.9</td>
</tr>
<tr>
<td>AAF (Ke et al. 2018)‡</td>
<td>ResNet101</td>
<td>79.1</td>
</tr>
<tr>
<td>PSANet (Zhao et al. 2018)‡</td>
<td>ResNet101</td>
<td>80.1</td>
</tr>
<tr>
<td>DFN (Yu et al. 2018)‡</td>
<td>ResNet101</td>
<td>79.3</td>
</tr>
<tr>
<td>DepthSeg (Kong and Fowlkes 2018)‡</td>
<td>ResNet101</td>
<td>78.2</td>
</tr>
<tr>
<td>DenseASPP (Yang et al. 2018)‡</td>
<td>DenseNet161</td>
<td>80.6</td>
</tr>
<tr>
<td>SVCNet (Ding et al. 2019)‡</td>
<td>ResNet101</td>
<td>81.0</td>
</tr>
<tr>
<td>DANet (Fu et al. 2018)‡</td>
<td>ResNet101</td>
<td>81.5</td>
</tr>
<tr>
<td>GFFNet (Ours)‡</td>
<td>ResNet101</td>
<td>82.3</td>
</tr>
</tbody>
</table>

Table 6: State-of-the-art comparison experiments on ADE20K validation set. Our models achieve top performance measured by both mIoU and pixel accuracy.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>mIoU(%)</th>
<th>Pixel Acc.(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RefineNet (Lin et al. 2017a)</td>
<td>ResNet101</td>
<td>40.20</td>
<td>-</td>
</tr>
<tr>
<td>PSPNet (Zhao et al. 2017)</td>
<td>ResNet101</td>
<td>43.29</td>
<td>81.39</td>
</tr>
<tr>
<td>PSANet (Zhao et al. 2018)</td>
<td>ResNet101</td>
<td>43.77</td>
<td>81.51</td>
</tr>
<tr>
<td>EncNet (Zhang et al. 2018a)</td>
<td>ResNet101</td>
<td>44.65</td>
<td>81.69</td>
</tr>
<tr>
<td>GCNNet (Li and Gupta 2018)</td>
<td>ResNet101</td>
<td>44.81</td>
<td>81.19</td>
</tr>
<tr>
<td>GFFNet (Ours)‡</td>
<td>ResNet101</td>
<td>45.33</td>
<td>82.01</td>
</tr>
</tbody>
</table>

Figure 6: Visualization of learned gate maps on ADE20K dataset. $G_i$ represents the gate map of the $i$th layer. Best view in color and zoom in for detailed information.

**Visualization of Gates**

In this section, we visualize what gates have learned and analyze how gates control the information propagation. Fig 6 shows the gates learned from ADE20K and Fig 7(a) shows the gates learned from Cityscapes respectively. For each input image, we show the learned gate map of each level. As expected, we find that the higher-level features (e.g., $G_3$, $G_4$) are more useful for large structures with explicit semantics, while the lower-level features (e.g., $G_1$ and $G_2$) are mainly useful for local details and boundaries.

Functionally, we find that the higher level features always spread information to other layers and only receive sparse feature signals. For example, the gate from stage 4 (in $G_4$ of Fig 7) shows that almost all pixels are of high-confidence. Higher-level features cover a large receptive field with fewer details, and they can provide a ground scope of the main semantics.

In contrast, the lower level layers prefer to receive information while only spreading a few sparse signals. This verifies that lower level representations generally vary frequently along the spatial dimension and they require additional features as semantic supplement, while a benefit is that lower features can provide precise information for details and object boundaries ($G_2$ in Fig 6 and $G_1$ in Fig 7(a)).

To further verify the effectiveness of the learned gates, we set the value of each gate $G_i$ to zero and compare the segmentation results with learned gate values. Fig 7(b) shows the comparison results, where wrongly predicted pixels after setting $G_i$ to zero are highlighted. Information through $G_3$ and $G_2$ is mainly help for object boundaries, while information through $G_1$ and $G_2$ is mainly help for large patterns such as cars. Additional visualization examples for the gates can be found in the supplementary materials.

**Results on Other Datasets**

ADE20K is a challenging scene parsing dataset annotated with 150 classes, and it contains 20K/2K images for training and validation. Images in this dataset are from more different scenes with more small scale objects, and are with...
Table 7: Per-category results on Cityscapes test set. Note that all the models are trained with only fine annotated data. Our method outperforms existing approaches on 15 out of 19 categories, and achieves 82.3% mIoU.

Table 8: Results on Pascal Context testing set.

Table 9: Results on COCO stuff testing set.

Conclusion

In this work, we propose Gated Fully Fusion (GFF) to fully fuse multi-level feature maps controlled by learned gate maps. The novel module bridges the gap between high resolution with low semantics and low resolution with high semantics. We explore the proposed GFF for the task of semantic segmentation and achieve new state-of-the-art results on four challenging scene parsing dataset. In particular, we find that the missing low-level features can be fused into each feature level in the pyramid, which indicates that our module can well handle small and thin objects in the scene.
Huang, G.; Liu, Z.; van der Maaten, L.; and Weinberger, K. Q. Neural computation
Hochreiter, S., and Schmidhuber, J. 1997. Long short-term mem-
Hariharan, B.; Arbelaez, P.; Girshick, R.; and Malik, J. 2015. Hy-
segmentation of small objects and modeling of uncertainty in urban
remote sensing images using deep convolutional neural networks. In CVPRW.
affinity fields for semantic segmentation. In ECCV.
Kong, S., and Fowlkes, C. C. 2018. Recurrent scene parsing with perspective understanding in the loop. In CVPR.
Li, Y., and Gupta, A. 2018. Beyond grids: Learning graph represen-
tations for visual recognition. In NeurIPS.
Liang, X.; Zhou, H.; and Xing, E. 2018. Dynamic-structured se-
matic propagation network. In CVPR.
objects in context. In European conference on computer vision, 740–755. Springer.
Lin, G.; Milan, A.; Shen, C.; and Reid, I. D. 2017a. Refinenet: Multi-path refinement networks for high-resolution semantic seg-
mentation. In CVPR.
Lin, T.-Y.; Dollár, P.; Girshick, R. B.; He, K.; Hariharan, B.; and Belongie, S. J. 2017b. Feature pyramid networks for object detec-
tion. In CVPR.
ECCV.
Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional
networks for semantic segmentation. In CVPR.
detection and semantic segmentation in the wild.
Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito,
Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Auto-
matic differentiation in pytorch. In NIPS-W.
Peng, C.; Zhang, X.; Yu, G.; Luo, G.; and Sun, J. 2017. Large
kernel matters – improve semantic segmentation by global convolu-
tional network. In CVPR.
Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-net: Convolu-
tional networks for biomedical image segmentation. MICCAI.
Srivastava, R. K.; Greff, K.; and Schmidhuber, J. 2015. Training
very deep networks. In NIPS.
Wang, X.; Girshick, R.; Gupta, A.; and He, K. 2018. Non-local
neural networks. In CVPR.
Xu, H.; Gao, Y.; Yu, F.; and Darrell, T. 2017. End-to-end learning
of driving models from large-scale video datasets. In CVPR.
Xu, D.; Ouyang, W.; Wang, X.; and Sebe, N. 2018. Pad-net: Multi-
tasks guided prediction-and-distillation network for simultaneous
depth estimation and scene parsing. In CVPR.
Yang, M.; Yu, K.; Zhang, C.; Li, Z.; and Yang, K. 2018. Densesapp
for semantic segmentation in street scenes. In CVPR.
Learning a discriminative feature network for semantic segmenta-
tion. In CVPR.
Zhang, H.; Dana, K.; Shi, J.; Zhang, Z.; Wang, X.; Tyagi, A.; and
Zhang, Z.; Zhang, X.; Peng, C.; Xue, X.; and Sun, J. 2018b. Ef-
xuf: Enhancing feature fusion for semantic segmentation. In
ECCV.
parsing network. In CVPR.
Zhao, H.; Zhang, Y.; Liu, S.; Shi, J.; Change Loy, C.; Lin, D.; and
parsing. In ECCV.