Searching for Contextual Subtasks for Semantic Segmentation

by

Connor McLaughlin

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Computer Science

by

April 28th 2022

APPROVED:

Professor Jake Whitehill, Thesis Advisor

Professor Emmanuel Agu, Reader

Professor Craig Wills, Head of Department

Abstract

Deep learning approaches for semantic segmentation have achieved tremendous success through their ability to model long-range scene context. However, by training for per-pixel classification, these methods fail to address the issue of class imbalance in segmentation datasets. Our work approaches segmentation from a contextual modeling standpoint by introducing novel subtasks as a human-provided hint or an auxiliary training signal. We first validate candidate subtasks through human-in-the-loop techniques to correct mistakes in segmentation, improving the mIoU of UPerNet-ResNet50 from 42.05 to 48.70 without any trained parameters. We then demonstrate the potential for multi-task learning of these subtasks with segmentation through a study of task gradients and end-to-end training.

Acknowledgements

I would like to thank Professor Jacob Whitehill for his invaluable mentorship. His patience and support encouraged me through each step of the research process, as well as towards a future career in research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The last decade has overseen a renaissance for computer vision. Large labeled datasets and increased computing power have allowed deep learning methods to surpass traditional computer vision techniques for various tasks, such as image classification, object detection, and semantic segmentation.

Semantic segmentation predicts a class label for each pixel in the input image. It boasts a wide range of applications, including medical imaging, autonomous driving, and scene parsing. Each target domain brings a unique set of challenges: medical imaging demands high-resolution precision, autonomous driving requires real-time inference, and scene parsing requires long-range contextual modeling across a large vocabulary of object classes. The focus of our work is on the latter task of scene parsing. Figure 1.1 shows an example input image and semantic segmentation label on the ADE20k [26] dataset.

Most recent work in scene parsing has focused on modeling long-range context. Following ParseNet [12], many approaches [2, 11, 20, 25] implicitly improve scene modeling by expanding the receptive field of the network. However, these methods train using a per-pixel classification loss, which may lead to class imbalance for less

(a) Scene Image (b) Segmentation Label

Figure 1.1: Image of an indoor scene from ADE20k.

frequent classes and those that consist of fewer pixels. Furthermore, traditional methods for handling class imbalance, such as weighted losses [1], are proven ineffective on scene parsing datasets [19].

We seek to solve the issue of class imbalance for scene parsing by incorporating a novel set of contextual subtasks into the network. These contextual subtasks contain high-level information robust to class pixel frequencies, such as the list of classes present in the image. We then explore two methods for integrating this context with existing networks: first, we pass it to the network as a hint, such as from a human annotator, and evaluate its efficacy using human-in-the-loop techniques to correct trained segmentation networks. Second, we propose these sources of context as secondary training objectives, thereby promoting the extraction of scene context without the biases of per-pixel classification. We validate our hypothesis by studying the gradient similarity for each task before modifying existing approaches to train jointly for these subtasks with segmentation. In contrast to other multi-task works

[20, 22], our auxiliary subtasks do not require additional annotations beyond the original segmentation label or contain extra parameters. Thus our work is compatible with most segmentation architectures, offering a promising alternative to the standard per-pixel classification objective.

1.1 Outline

The structure of our work is as follows: Section 2 introduces relevant background on semantic segmentation, human-in-the-loop deep learning, and multi-task learning methods. Section 3 outlines the foundations for our methods, with sections 3.1 and 3.2 covering aspects for human-corrected segmentation and multi-task learning, respectively. We share our experimental results in section 4, and discuss their impact in section 5.

Chapter 2

Background

We identify four key areas in literature for semantic segmentation. First, we review advancements in segmentation architectures, followed by previous works for handling class imbalance. We then introduce seminal works for human-in-the-loop and multi-task segmentation.

2.1 Segmentation Architectures

Most recent work in segmantic segmentation is based on the Fully Convolutional Networks (FCN) [14] architecture. The baseline FCN adapts image classification networks (e.g. CNNs such as ResNet [8]) for dense prediction through a two-stage *encoder-decoder* framework. The *encoder* stage extracts a feature representation from the image using the backbone CNN without the final fully-connected layers. This backbone is often pretrained on large-scale classification datasets such as ImageNet [5], leveraging benefits of transfer learning. The *decoder* stage transforms the feature representation to the resolution of the input image (i.e. through bi-linear interpolation) to produce per-pixel classifications.

In this approach, the image context is gradually obtained by stacking convolution,

striding, and pooling layers that widen the backbone network's receptive field. However, as found by Liu et al. in ParseNet [12], the empirical receptive field of the FCN encoder is not large enough to capture global image context. This motivated subsequent works to approach semantic segmentation through the lens of improving the empirical receptive field to promote the use of global image context.

At the encoder stage, there are two common approaches for improving receptive field. First, convolutions in the backbone may be replaced by dilated convolution [21]. This is often accompanied by the removal of downsampling steps, resulting in a higher spatial resolution and a larger receptive field. More recently, vision transformers [6, 13], which capture long-range relations from the first layer, have successfully replaced CNN backbones.

At the decoder stage, the most common way to improve the receptive field is by building image feature pyramids. Feature Pyramid Network (FPN) [11] creates multi-scale representations through the lateral connections between each spatial scale of the encoder backbone. Pyramid Scene Parsing Network (PSPNet) [25] builds a feature pyramid by applying pooling kernels of various sizes to the backbone feature map, and concatenating the resulting representations. Other approaches, such as OCNet [23] and CCNet [10] utilize attention mechanisms to aggregate wide-range object context. The most prevalent architecture in recent benchmarks is UPerNet [20], which combines techniques of FPN and PSPNet to aggregate context. Contrary to previous methods, UPerNet is not dependent on dilated encoder backbones, which results in faster and more memory-efficient training.

2.2 Handling Class Imbalance

While methods that address the receptive field of the network have achieved notable success in end segmentation accuracy, few works have tackled the contextual modelling of scenes with attention to class imbalance. EncNet [24] re-weighs the final feature map of the encoder with a separate branch trained with an image-level classification loss. This task predicts whether there is at least one pixel present in the image for each class. In contrast to segmentation loss, this image-level classification task treats small and large objects equally. We compare each loss function in Equation 2.1. Note that in segmentation, y consists of a single class label for each pixel, while in classification y consists of a single vector for the entire image, and multiple classes may be present.

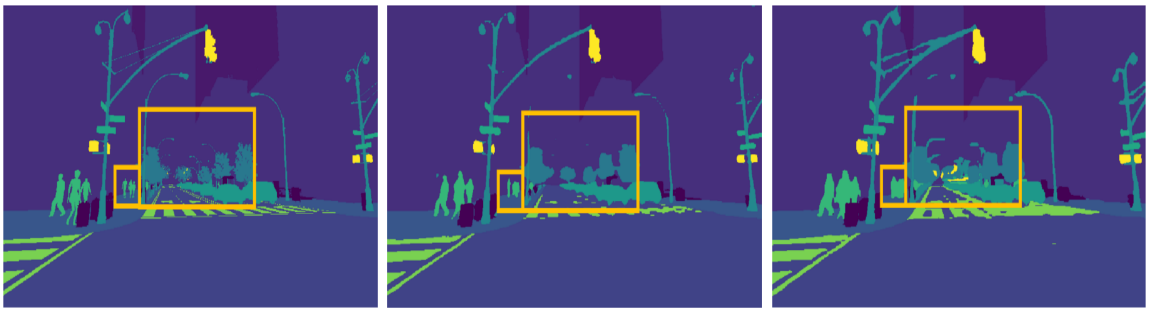
$$L_{seg}(y, \hat{y}) = \frac{1}{W \times H \times C} - \sum_{i=0}^{W} \sum_{j=0}^{H} \sum_{k=0}^{C} y_{(i,j)}^{k} log(\hat{y}_{(i,j)}^{k}) \tag{2.1a}$$

$$L_{cls}(y, \hat{y}) = -\sum_{k=0}^{C} y^{k} log(\hat{y}^{k}) \tag{2.1b}$$

In 2022, MaskFormer [3] avoids per-pixel classification loss altogether by re-framing semantic segmentation as a mask prediction and classification problem. This method yielded the largest performance increase relative to per-pixel classification methods on datasets with a large number of classes.

Other approaches have tried to alleviate class imbalance by re-weighing per-pixel cross-entropy losses. The most common approach is known as *inverse frequency weighing*, which weighs the loss for each class inversely to its frequency relative to other images across the training set. A pioneer work in segmentation, SegNet [1],

proposed *median frequency balancing*, which weighs pixels relative to the median class' proportion of pixels present in images. However, while effective on datasets with fewer classes (≤ 20), this method is not common practice on large vocabulary scene parsing datasets as it often harms overall accuracy [19]. One explanation is that the large quantity of infrequent object classes obtain too much weight and tend to be over-segmented. An example of this effect is shown in Figure 2.1.



(a) Target segmentation label. (b) Prediction of model trained with cross entropy loss (under-segmented). (c) Prediction of model trained with weighted cross entropy loss (over-segmented).

Figure 2.1: Segmentation errors for models trained with weighted and unweighted per-pixel cross entropy. Figure taken from [19].

A more recent approach [19] instead weighs losses using an online hard-mining regime that weighs classes based on their recall performance rather than their dataset frequency. While this technique obtains more robust results than previous methods, they found that large vocabulary datasets do not benefit from this method, due to the presence of many classes with low recall.

2.3 Human-in-the-loop

As recent advancements in semantic segmentation networks are prone to making mistakes, especially on smaller object classes, a field of research for human-guided segmentation has emerged. One possible application of a collaborative human-

machine segmentation model is in fast annotation of new segmentation datasets, in which a pretrained model provides an initial estimate of the segmentation label, and the human refines the prediction. This can greatly improve the speed of per-pixel annotation.

One recent application in semantic segmentation is GuideMe [17], which uses natural-language hints from a human in order to correct segmentation predictions. This method uses a Recurrent Neural Network (RNN) to encode the hint, which then modulates the intermediate feature maps of the segmentation network.

For feature modulation, GuideMe uses a modification of Feature-wise Linear Modulation (FiLM). Rather than predicting a scaling and bias term for each spatial location of the feature map, X, FiLM uses a single scaling and bias term (denoted as γ^s and γ^b) for each channel of the feature map (see Equation 2.3.1: FiLM). This prevents overfitting, as vision models have dense feature representations which would require millions of parameters to modulate element-wise (e.g. a 128x128x2048 feature map in UPerNet corresponds to 67M parameters for element-wise reweighing). As natural language hints to GuideMe [17] may provide spatial information (e.g. *there is a cloud in the top of the image*), they add additional terms α and β to FiLM which apply different weights at each location in the feature map (Equation 2.3.2: GuideMe). Additionally, they add a vector of ones to the scaling term, such that the feature modulation results in the identity mapping with zero-initialized weights.

$$X'_c = \gamma^s_c X_c + \gamma^b_c \qquad \text{(2.3.1: FiLM)}$$

$$X'_{w,h,c} = (1 + \gamma^s_c + \alpha_w + \beta_h) X_{w,h,c} + \gamma^b_c \qquad \text{(2.3.2: GuideMe)}$$

2.4 Multi-task Segmentation

Another active branch of research in semantic segmentation leverages multi-task learning to promote data efficient training and to learn a more robust representation shared across tasks. Tasks commonly grouped with semantic segmentation include depth estimation [16], surface normal prediction [18], and object part segmentation [20]. However, training for these tasks requires additional data beyond the segmentation label, which may not be readily available.

The most straightforward design for multi-task learning is to share the same backbone encoder across tasks, with separate decoder heads for each task. Thus each decoder head is guided solely by its task-specific loss, while the encoder is guided by all of the task losses. Even this simple approach can lead to significant improvements [20, 22] for segmentation, indicating the benefits of robust multi-task representations.

Chapter 3

Proposed Methods

In this chapter we first introduce potential contextual subtasks and their immediate application for human-in-the-loop correction (section 3.1), and then we propose a framework for training jointly for these tasks and segmentation (section 3.2).

3.1 Contextual Subtasks

3.1.1 Image-Level Classification

The most straightforward task is *image-level classification*, or predicting the list of objects present at all in the segmentation label, as used in EncNet [24]. This takes the form of a vector $(0, 1)^C$ containing independent binary labels for each class. This context informs the network of which labels it should add or remove from the prediction.

We note that this context can be used to modulate the outputs of the network at any hidden layer up to the final prediction space. However, as image-level classification hints contain high-level information, we propose that the feature modulation should occur at the end of the segmentation network. This intuition is also supported by

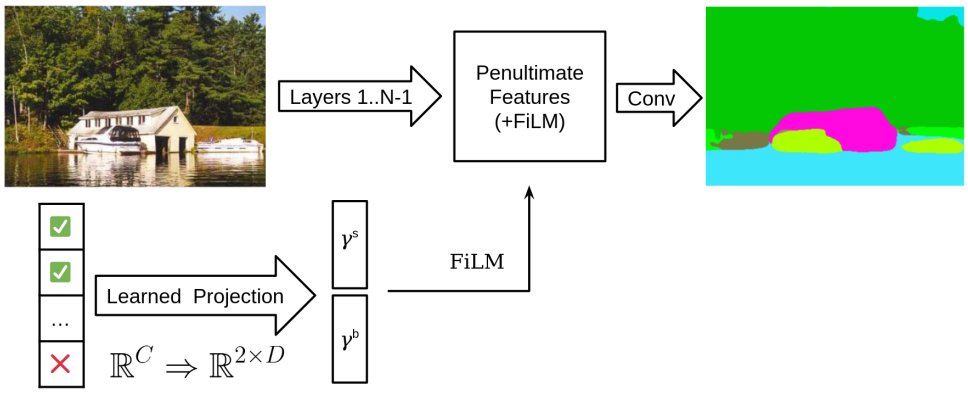


Figure 3.1: FiLM Modulation Pipeline

the experimental results of GuideMe [17]. For simplicity, we consider only the penultimate feature map (i.e. immediately preceding the classification layer), as this contains linearly separable information on each class and contains more information than the predicted logits.

To use the image-level classification vector to modulate intermediate features, we must first align the channel dimensions of our provided context and the target layer. To do so, we apply a multi-layer perceptron (MLP) to project our class vector from $\mathbb{R}^C$ to $\mathbb{R}^{2D}$, where D is the number of feature channels. The first D components of our projected embedding are used as the scaling term for FiLM, and the remaining D as the bias term. This pipeline is illustrated in Figure 3.1.

3.1.2 Sub-Region Classification

As image-level classification does not contain localized information, we propose sub-region classification, which predicts an independent list of object classes for each region. If we partition our image into k regions, then extract class vectors in $\mathbb{R}^C$ for each region, note that the result is a $\mathbb{R}^{k\times k\times C}$ context label. We project each spatial location independently (i.e. 1x1 through 1x1 convolution), and apply the resulting FiLM terms to each corresponding region. We note that sub-region classification

is equivalent to segmentation when the number of regions k is equal to the image resolution.

3.1.3 Distribution Information

While previous hints address the concern of what classes are present and where, they do not indicate how many pixels belong to each label. We propose pixel distribution learning as an extension to the previous classification task. Rather than giving a binary label of whether or not each class is present, we provide the proportion of pixels assigned to each class within the region. As with classification hints, we may project the corresponding context vectors and modulate our features with FiLM. However, we note that distribution knowledge is enough to correct predictions without trained parameters: given segmentation output $\hat{y}^{WxHxC}$ containing our softmax predictions for each pixel in the image, we can extract the predicted pixel distribution for a region through average pooling. For example, at the image level, we extract the predicted distribution vector $\hat{d} \in \mathbb{R}^C$. We then offset our softmax distribution to by the difference in d and $\hat{d}$: $\hat{y}_{corrected} = \hat{y} + (d - \hat{y})$.

3.1.4 Top-K Hints

For the application of human-in-the-loop segmentation, each task should possess two fundamental properties: first, they should be simple to implement with various architectures, and second, they should require little work from the human annotator.

Our proposed subtasks offer a trade-off of these qualities, as classification is straightforward for human annotators, even at the sub-region scale, but requires trained FiLM parameters to correct existing networks. Distribution information requires much more detailed labelling from humans but may be used to adjust the output of any network without any trained parameters.

We note that in the case of distribution hints especially, the amount of effort required to capture the proportion of pixels for each class may be prohibitively difficult for annotators. Therefore, we propose a simplified *top-k* hint, in which the annotator provides only the classification or distribution information for the k most prominent classes in the scene. With top-k hints, our FiLM correction method is unchanged, but in our parameter-free distribution correction, we set the correction term for classes outside the top-k to be 0.

3.2 Multi-Task Semantic Segmentation

We begin our discussion of multi-task learning with several key observations about the previous contextual subtasks. First, recall that image-level classification avoids class imbalance caused by pixel frequencies, as all classes with ≥ 1 pixel are labelled as present. If we partition our image for sub-region classification, we sacrifice some of this *pixel frequency invariance* in our loss, as larger objects may be present in multiple sub-regions, however this task is more similar to segmentation.

Furthermore, the task of sub-region classification possesses the qualities of a recall-based loss weighting method. As discussed in [19], it is important for loss weighting regimes to weigh classes based on their difficulty to predict, rather than their prevalence in the training set. Likewise, the magnitude of our sub-region classification loss is proportional to the networks ability to detect whether a class is present at all in the region, rather than being based on its ability to predict individual pixels.

With these factors in mind, we propose a simple hypothesis:for a given segmentation architecture and target dataset, *there exists a combination of sub-region classification loss scales which outperform a network trained with standalone per-pixel*

classification loss.

3.2.1 Multi-Task Architectures

We begin our study of multi-task architectures by reviewing the task of per-pixel classification. The prediction head for most FCN-based approaches is as follows: given a penultimate feature map $X \in \mathbb{R}^{\frac{W}{s} \times \frac{H}{s} \times D}$, per-pixel classifications are obtained by 1x1 convolution followed by bi-linear interpolation and softmax activations $\hat{y} \in \mathbb{R}^{W \times H \times C}$. Therefore, each spatial location in the feature map contains linearly separable information for the classes of corresponding pixels.

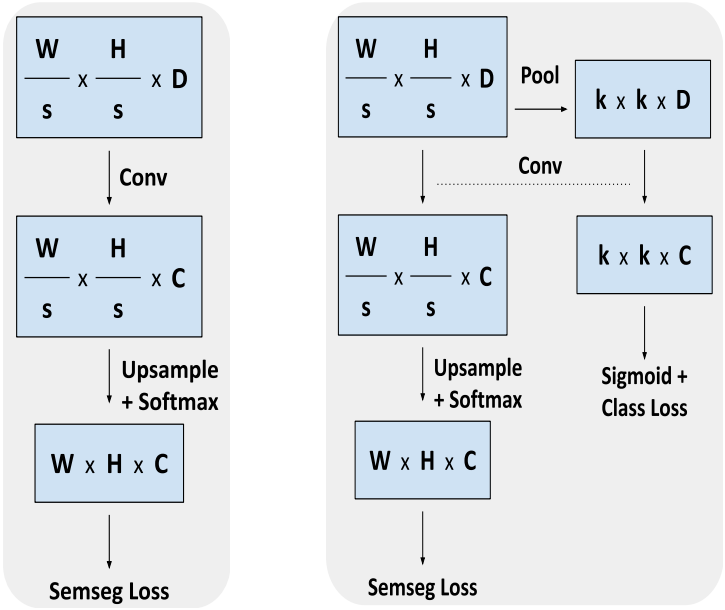
With this in mind, it is reasonable to assume that the average feature over a region of the image should contain enough information to perform sub-region classification. Therefore, our multi-task architecture for classification consists of average pooling of the penultimate feature map to the desired sub-region scale, followed by a 1x1 convolution layer with logistic sigmoid activations. In our work, we share the weights for 1x1 convolution layers between our classification and segmentation tasks. We train with binary cross-entropy loss applied independently to each channel and spatial location in the output.

The prediction for the pixel distribution subtask may be obtained through average pooling across per-pixel softmax outputs. However, this training objective does not have any advantages with respect to class imbalance compared to segmentation, and therefore we do not include it in our multi-task training experiments.

We scale how much segmentation and classification loss with a single parameter α, shown in Equation 3.1.

$$L_{mtl}(L_{seg}, L_{cl}, \alpha) = (1 - \alpha) \times L_{seg} + \alpha \times L_{cl} \tag{3.1}$$

We illustrate the architectures for multi-task learning in Figure 3.2.



(a) Baseline (b) Multi-Task

Figure 3.2: Baseline and Multi-task Loss Architectures.

3.2.2 Handling Conflicting Tasks

When training multiple tasks simultaneously, it is possible for the gradients of each task to point in different directions. This occurrence of *conflicting gradients* has been well-reviewed in literature. One potential problem of conflicting gradients is that their combined gradient magnitude may be much smaller to that of the respective tasks, leading to slower model training.

Several methods have been proposed to handle conflicting gradients. Du et al. [7] set the gradient of auxiliary tasks to zero if the cosine similarity with the primary task is negative. PCGrad [22] avoids removing conflicting gradients entirely by instead removing the projection of the conflicting gradients from each other, thus ensuring a positive cosine similarity. However, conflicting gradients may not be inherently

harmful. In our application, this might indicate that the per-pixel classification loss moves our weights to be biased towards prominent classes, while the image-level classification loss goes against this bias. Therefore, we do not take any measures to remove negate the direction of conflicting gradients.

Additionally, to more direct comparison with standalone per-pixel classification loss, we scale our multi-task loss to have the matching gradient magnitude as segmentation. This allows us to attribute any changes in model performance solely to the shifted gradient direction achieved through introducing additional tasks.

Chapter 4

Experiments and Results

We present our results as follows: in section 4.1 we review implementation details for our experiments, in section 4.2 we describe our results for human-in-the-loop segmentation for each contextual subtask. In section 4.3.2 we describe our results for mutli-task training experiments, and explore the underlying mechanism behind our model.

4.1 Experimental Setup

4.1.1 Dataset

We use the challenging ADE20k [26] scene parsing dataset in our experiments. ADE20k consists of 20k training samples and 2k validation samples containing 150 unique object classes. As is common in literature, we use the mean intersection over union (mIoU) metric, which is computed as the average intersection over union for each label class, ignoring pixels labeled as "background".

All of our experiments utilize the UPerNet [20] decoder head, due to its prevalence in literature and training efficiency. For backbones, we explore both CNN and

Transformer-based models. Following public benchmarks [4], we use a ResNet50 variant with the first 7x7 convolution kernel replaced by 3 3x3 convolutions. We additionally use the Swin-Tiny [13] for a reference transformer backbone. Pretrained models come from the mmsegmentation [4] public benchmark.

ResNet-based models are trained using SGD with an initial learning rate of 0.01, and momentum of 0.9. Swin-based models are trained using AdamW [15] with an initial learning rate of 0.0001. All models are trained with a weight decay of 5e-4. For data augmentation, we perform the standard random scale in the range [0.5, 2], random horizontal flips, color jittering, and random cropping. We train all models using crop sizes of 512x512. All mini-batch sizes are set to 16.

For human-in-the-loop segmentation, we train for a 5K iterations with a constant learning weight. For end-to-end multi-task training, we train for 160K iterations using a *polynomial* learning rate scheduler: $LR = (1 - (\frac{iter}{iter_{max}})^{0.9})$.

When reporting mIoU, we list single-scale results without any test-time augmentation.

4.2 Human-in-the-loop Segmentation

4.2.1 Classification Hints

For classification hints, we evaluate the mIoU improvements for a trained segmentation model with varying projection heads and sub-region scales.

Ablation Study for Projector Head. We evaluate the accuracy improvement of our model given ground-truth image-level ($k = 1$) classification labels, with varying projector head complexity. As shown in Table 4.1, we find that there is no benefit to a more complex projection head, and thus our future experiments default to a linear projection from $\mathbb{R}^C \Rightarrow \mathbb{R}^D$.

Table 4.1: Performance improvement using ground-truth classification labels for FiLM modulation with varying projection heads.

Backbone	mIoU (s.s.)	Projection Layers
ResNet50	42.05	N/A (baseline)
	50.18 (+8.13)	1
	50.22 (+8.15)	2

Sub-Region Classification. We additionally evaluate FiLM + classification labels for region scales $k > 1$. Clearly, as k increases, our accuracy will increase, but as shown in Table 4.2 we find that even coarser scales are effective at improving model mIoU. This indicates that the signal from image-level classification is relevant for end segmentation.

Table 4.2: UPerNet performance given sub-region class labels with FiLM modulation.

Backbone	mIoU	Region Scale
ResNet50	42.05	N/A (baseline)
	50.18 (+8.13)	1x1
	55.55 (+13.50)	2x2
	59.31 (+17.26)	3x3

4.2.2 Distribution Hints

For distribution hints, we focus on the parameter-free correction method, and evaluate the efficacy of parameter-free correction with varying $top - k$ labels and sub-region scales.

Top-k Distribution Hints. We show our results using top-k distribution labels for varying k in Table 4.3. We note that even values of k as small as 3 improves model mIoU.

Sub-region Distribution Hints. We evaluate the benefit for including distribution hints for varying scale sub-regions, using $k = 150$. As expected, Table 4.4 shows

Table 4.3: UPerNet performance given distribution labels with parameter-free correction.

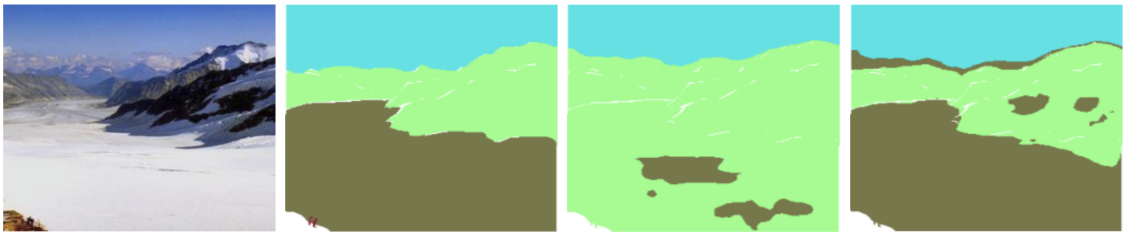
Backbone	mIoU	Distribution K
ResNet50	42.05	N/A (baseline)
	48.70	150 (all classes)
	44.16	5
	42.98	3
	41.93	2

substantial benefits for finer-resolution hints.

Table 4.4: UPerNet performance given distribution labels for multiple sub-region scales.

Backbone	mIoU	Distribution Scale
ResNet50	42.05	N/A (baseline)
	48.70	1x1
	56.28	2x2
	61.66	3x3
	70.97	6x6

An example segmentation correction given an image-level distribution hint is shown in Figure 4.1. Even without local context, our method successfully redistributes misclassified pixels to the desired label.



(a) Input image (b) Segmentation Label (c) Predicted segmentation (d) Corrected Segmentation

Figure 4.1: Sample corrected segmentation using distribution hints.

4.2.3 Discussion

Our results show the promise of both classification and distribution subtasks for human-in-the-loop segmentation. Our methods may be integrated into any off-the-shelf segmentation network, requiring at most the training of a single linear layer to correct predictions. Additionally, we offer a convenient trade-off between annotator workload and improved accuracy with sub-region and top-k alternatives for each hint. However, our experiments assume perfect annotator accuracy, and do not explore the behavior of each method when given inaccurate hints.

4.3 Multi-Task Segmentation

4.3.1 Analysis of Task Gradients

We first validate our multi-task learning hypothesis by examining the cosine similarity of segmentation and varying sub-region classification tasks. We use a pretrained (for segmentation alone) UPerNet-Swin model and take the gradient of the final convolution layer. Figure 4.2 shows the gradient similarity of each task aggregated over the first 500 samples. As anticipated, the classification gradients become more similar to segmentation as we increase the number of sub-regions. Our classification tasks are also consistently more similar to segmentation than inverse-frequency weighted segmentation. This indicates that our classification loss carries a different type of training signal than simple loss weighting.

4.3.2 End-to-end Training

Ablation Study for Loss Weight. We train UPerNet models with ResNet50 backbones with varying classification loss proportions. Our results in Table 4.5

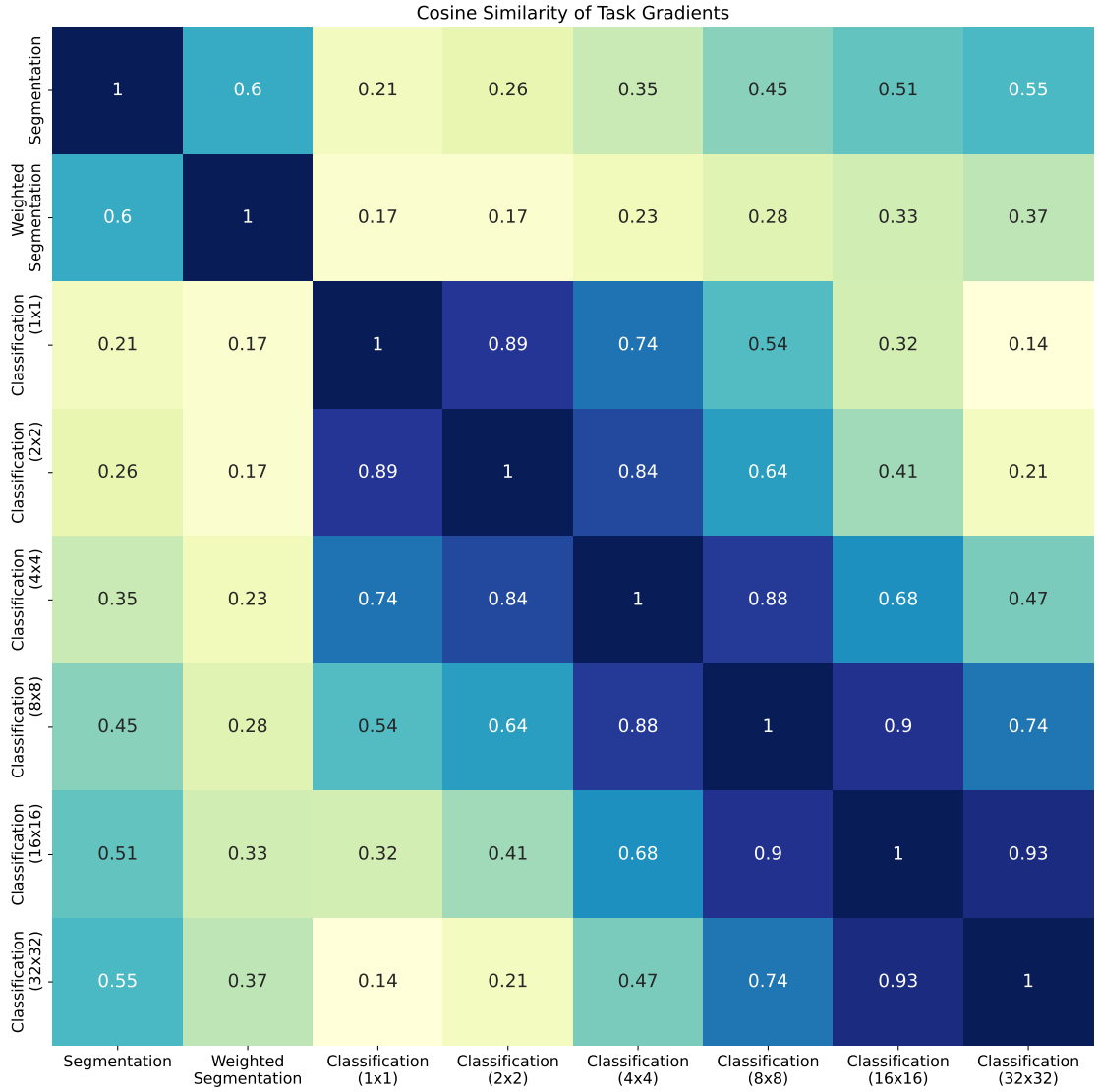


Figure 4.2: Gradient Cosine Similarity for Segmentation and Classification Tasks.

indicate that $\alpha = 0.25$ is empirically the best trade-off between segmentation and classification loss.

Table 4.5: Performance vs. Loss Weight using UPerNet with ResNet50 backbone. All models use image-level classification losses.

Loss Weight α	MIoU
Baseline ($\alpha = 0$)	42.05
$\alpha = 0.125$	43.88
$\alpha = 0.25$	**44.61**
$\alpha = 0.5$	43.27

Method efficacy. We explore the benefit of our multi-task loss on both ResNet-50 and Swin-T and compare to existing benchmarks. Notably, we find that our multi-task ResNet-50 model outperforms a segmentation-only model with a deeper ResNet-101 backbone. However, on the more recent Swin-T backbone, we find our additional loss term has a slight negative effect on overall accuracy.

Table 4.6: Multi-loss performance on multiple backbones. All our models use a $\alpha = 0.25$. Baselines for segmentation-only models from mmsegmentation [4].

Model	Backbone	Loss	MIoU
UPerNet [20]	ResNet50	SL	42.05
	ResNet101	SL	43.82
	ResNet50	SL + CL	44.61 (+2.56)
	Swin-T [13]	SL	44.41
	Swin-T	SL + CL	43.99 (-0.43)

We propose two possible explanations for the discrepancy between ResNet and Swin performance with multi-task learning. The first is that the transformer architecture is inherently better at capturing smaller object classes than convolution, and therefore there is not as much to gain. Another possibility is that our multi-task loss functions primarily as a form of regularization. Additional regularization may benefit ResNet50 more than Swin, as the latter already utilizes techniques such as stochastic depth [9].

4.3.3 Discussion

Our work with multi-task learning is most similar to EncNet [24], which uses image-level classification as a supervisory signal. However, our work differs by focusing on the effect of the multi-task loss itself rather than architectural changes. While our multi-task objective did not always yield improved accuracy, our study of gradients indicates that more fine-grained sub-region scales may be more compatible with the goal task of segmentation, and these objectives still provide benefits over standalone segmentation for handling class imbalance.

One caveat of our multi-task loss is in the number of hyperparameters, including the loss weight, strategy for handling conflicting gradients, and sub-region scales for classification. The optimal value for the latter is also likely to depend on the size and difficulty of object classes, requiring individual tuning for each segmentation dataset.

Chapter 5

Conclusion

5.1 Overview

In this work, we addressed the issue of class imbalance in semantic segmentation through the introduction of contextual subtasks. We first evaluated the efficacy of our sub-region classification and distribution hints as provided in a human-in-the-loop setting. These hints offer a simple trade-off between annotation detail and model performance. In addition, they can be easily integrated into existing architectures as they require at most one linear layer to train. Without any additional parameters, our distribution correction method improves the existing UPerNet-ResNet50 model from 42.05 to 48.70 mIoU. We also demonstrate the potential of our classification subtasks as a multi-task training objective through a comparison of task gradients. Our initial experiments on UPerNet-ResNet50 enhanced the mIoU from 42.05 to 44.61 through the addition of the image-level classification task. We believe that this method may also have success on more modern architectures with further exploration of classification scales for the auxiliary task.

Our proposed methods for human-in-the-loop and multi-task learning are model-

agnostic and widely applicable. In particular, our multi-task loss is a promising alternative for per-pixel classification loss. We hope that our experiments provide a foundation for future work in these areas.

5.2 Future Work

One immediate next step for both human-in-the-loop and multi-task methods is to evaluate their performance on other datasets. As pointed out by [19], the overall difficulty of ADE20k is so much greater than the other datasets that it is hard to attribute model performance to any specific factor, e.g. class frequency. Future studies may be able to examine the relationship between sub-region scales and performance improvements for each class.

5.2.1 Human-in-the-loop

For human-in-the-loop applications, the next step is to study human-computer interaction for our proposed contextual hints, notably our distribution hints. It is important to measure how accurate the estimate of the distribution needs to be and what coarseness of guidance ($top - k$, sub-region scale) is effective in practice. One potential experiment could measure the time saved in annotating new images by either starting from scratch, an existing model prediction, or an existing model corrected with our distribution hints.

5.2.2 Multi-Task Learning

In our experiments, we fixed the magnitude of the multi-task gradient to understand the effect of the gradient direction. Each trial should be repeated without this correction, as this may have limited our model. In practice, the difference in magnitude

between our multi-task and original gradient could function as an intelligent adaptive learning rate.

Future work should also explore different scales of classification subtasks. Our original hypothesis was that a beneficial class loss scale exists (between $k = 1$ and the image resolution), but our experiments only examined image-level classification due to time constraints.

Bibliography

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation, 2015.

[2] 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, 2016.

[3] B. Cheng, A. G. Schwing, and A. Kirillov. Per-pixel classification is not all you need for semantic segmentation, 2021.

[4] M. Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. `https://github.com/open-mmlab/mmsegmentation`, 2020.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[6] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020.

[7] Y. Du, W. M. Czarnecki, S. M. Jayakumar, M. Farajtabar, R. Pascanu, and B. Lakshminarayanan. Adapting auxiliary losses using gradient similarity, 2018.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[9] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Weinberger. Deep networks with stochastic depth, 2016.

[10] Z. Huang, X. Wang, Y. Wei, L. Huang, H. Shi, W. Liu, and T. S. Huang. Ccnet: Criss-cross attention for semantic segmentation, 2018.

[11] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection, 2016.

[12] W. Liu, A. Rabinovich, and A. C. Berg. Parsenet: Looking wider to see better, 2015.

[13] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021.

[14] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation, 2014.

[15] I. Loshchilov and F. Hutter. Decoupled weight decay regularization, 2017.

[16] A. Mousavian, H. Pirsiavash, and J. Kosecka. Joint semantic segmentation and depth estimation with deep convolutional networks, 2016.

[17] C. Rupprecht, I. Laina, N. Navab, G. D. Hager, and F. Tombari. Guide me: Interacting with deep networks, 2018.

[18] T. Standley, A. R. Zamir, D. Chen, L. Guibas, J. Malik, and S. Savarese. Which tasks should be learned together in multi-task learning?, 2019.

[19] J. Tian, N. Mithun, Z. Seymour, H.-P. Chiu, and Z. Kira. Striking the right balance: Recall loss for semantic segmentation. 2021.

[20] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun. Unified perceptual parsing for scene understanding, 2018.

[21] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions, 2015.

[22] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn. Gradient surgery for multi-task learning, 2020.

[23] Y. Yuan, L. Huang, J. Guo, C. Zhang, X. Chen, and J. Wang. Ocnet: Object context network for scene parsing, 2018.

[24] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal. Context encoding for semantic segmentation, 2018.

[25] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network, 2016.

[26] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba. Semantic understanding of scenes through the ade20k dataset, 2016.