Technische Universität Berlin

Institut für Softwaretechnik und Theoretische Informatik
Fachgebiet Neuronale Informationsverarbeitung

Fakultät IV
Marchstrasse 23
10587 Berlin

http://www.ni.tu-berlin.de

Master's Thesis

Introducing knowledge distillation in a multi-task neural network

Alejandro Hernández Munuera

Matriculation Number: 395678

Supervised by
Prof. Dr. Klaus Obermayer

Abstract

Multi-task Learning (MTL) consists of training a neural network (NN) to perform on more than one task exploiting the knowledge shared between tasks. One of the main problems in MTL is the lack of large datasets containing ground truth (GT) for several tasks on each of its data points. As a consequence, the loss and backpropagation process have to be adapted to this circumstance, undermining the performance of the model.

Based on previous work, we use a knowledge distillation (KD) technique to try to overcome this limitation and transfer a more general knowledge using other NN output. KD is a set up where a NN doesn't learn from the GT, instead, it extracts knowledge from another NN (already trained for the corresponding task) soft output.

Along this project, we analyze if the KD technique could be used to solve the data limitation that MTL faces and therefore exploit the MTL advantages in each training data point. Training a multi-task NN on the segmentation and object detection tasks, and making use of the VOC2012seg dataset, which has GT for both tasks on each data point, we run several experiments substituting GT of a specific task with the corresponding "tutor" output in different subsets of images.

The results show that KD and the appropriate "tutor" NN can allow MTL training process to lack 100% of the GT for a specific task and even reach a better performance than the same NN trained with the complete GT.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hifsmittel angefertigt habe.

Declaration of own work

I hereby declare that the thesis submitted is my own, unaided work, completed without any unpermitted external help. Only the sources and resources listed were used.

Berlin, May 1st, 2019

.....
Alejandro Hernández Munuera

Table of Contents

Figures, tables and formulas

1 Introduction

1.1 Context and motivation

In recent years, deep learning has developed as an important topic inside the machine learning and computer science fields. Among many purposes, NN have become one of the most important tools to process and understand images, where they will learn key features to generate the appropriate output. There are countless possible image processing tasks that can be approached with NN such as object detection and semantic segmentation among many others.

In many cases, the NN employed for different tasks have similar structures and configurations and even learn identical features. Due to this fact, in recent years MTL has been receiving more attention and research effort [27] [28] [30]. MTL neural networks process images and generate the output for several tasks exploiting the fact that many computer vision tasks share knowledge between them. Apart from offering this advantage, the methodology is significantly less computationally expensive than the classic one-task NN set up. MTL will result on having a single NN that has been trained once and can solve many tasks, while, with the classic one-task NN approach many NN will be needed including their respective and separated training process.

One of the main limitations that MTL faces is the lack of datasets containing GT for each task on every data point. As MTL is a recent methodology, most of the datasets are oriented to be used for one task only. Those cases where the dataset contains ground truth for several tasks, the number of images with ground truth for every task is usually low. As a consequence, the MTL will be forced to either use a small subset for training where every image contains the corresponding GT for each task or change the learning process in order to adapt to the lack of GT.

	ImageNet	**VOC 07**	**VOC 12**	**COCO**	**NYU**	**MSRA10K**
Detection	Partial	Yes	Yes	Yes	No	No
Semantic segmentation	No	Partial	Partial	Yes	Yes	No
Instance segmentation	No	Partial	Partial	Yes	No	No
Human parts	No	No	No	No	No	No
Human landmarks	No	No	No	Yes	No	No
Surface normals	No	No	No	No	Yes	No
Saliency	No	No	No	No	No	Yes
Boundaries	No	No	No	No	No	No
Symmetry	No	No	No	Partial	No	No

1 Datasets and the tasks GT contained [28]

Apart from MTL, is important for this project to introduce the concept of knowledge distillation. KD is a set up where a NN, "student", instead of learning from raw data, it learns from a combination of the GT and the soft output of a second NN, "tutor", that has already been trained for the corresponding task. This approach emerges a few years ago as a model compression technique [38] [36]. On this case, the "tutor" normally is a large NN that transfers the knowledge to a smaller one so it imitates its performance. Afterwards, this procedure was used to improve model performances relying on the fact that the combination of ground truth and the soft output of a "tutor" could offer a more complete and consistent information than raw data, allowing a better understanding of the task with the appropriate adaptations on the "student" loss function [35] [39].

1.2 Objective and scope

In this master thesis, we study how knowledge distillation could be a useful tool to overcome the MTL data limitation. For that purpose:

- We first gather the appropriate "tutors" for each of the tasks, and the MTL neural network to play the "student" role. In order to reproduce a transfer learning procedure where the MTL dataset lacks GT for a specific task and the "tutors" haven't seen these images before, the "tutors" needed to be trained on a different dataset from the one used for the learning process of the "student".

- The "tutors" output is treated and processed to be fed to the "student" NN. Different methods to combine raw data and "tutors" output are studied in order to obtain the best performance.

- Then, exploiting the fact that bigger datasets can be used for MTL making use of KD to generate the data needed to train the "student" NN, experiments are executed with different datasets to measure how beneficial and reliable KD could be.

1.3 Outline

In chapter 2 we review the work done in the past to develop MTL techniques and NN, and how its data limitation has been tackled. On the other hand, KD projects oriented to model compression and learning process improvements are described as well as their results. Besides, we explain fundamental concepts to understand the work and methodology employed on this master thesis.

In chapter 3, the methodology followed and the tools used along the thesis are presented. Then, in chapter 4 the corresponding experiments are explained and plots are shown presenting its results. To conclude, chapter 5 discusses the experiments, their results and present possible future lines of work.

2 Fundamentals and previous work

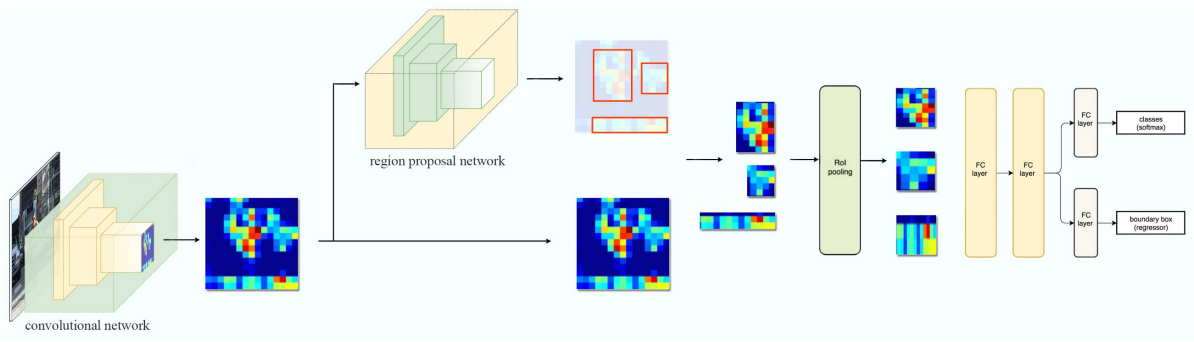
As the MTL neural network used in this project is trained on semantic segmentation and object detection, in the next points we explain the characteristics and some state-of-the-art projects for these two tasks.

2.1 Object detection

Object detection consists of perceiving the corresponding objects present on an image and generating a bounding box, a label and the confidence of the prediction for each object.

This deep learning topic has been well studied due to its many possible uses in different fields. As a result, several deep learning models have appeared with leading performance. Among them, the predominant projects used for object detection during the last years are Faster-R-CNN [7], SSD [9] and YOLO [10].

Faster R-CNN [7] is the result of the development of a region based neural network. This kind of neural network generates thousands of potential bounding boxes or Region of Interest (RoI) and process them to generate its corresponding confidence and class. It first appeared R-CNN [5], which showed great performance but low speed due to the fact that each RoI is processed and classified independently by a convolutional neural network and then filtered out based on confidence. Fast R-CNN [6] further improves performance and decrease computational cost performing the feature extraction once over the whole image instead of one time per RoI. Ultimately, Faster R-CNN [7] shrinks the processing time and introduce a new region proposal method obtaining the RoIs from the feature map through a neural network.



2 Faster R-CNN structure [3]

Apart from this project line, other projects such as OverFeat [8], SSD [9] and YOLO [10] have used sliding windows to generate RoIs reducing significantly the computational cost and keeping a state-of-the-art performance.

On this master thesis, we use Faster R-CNN [7] as the object detection "tutor" for a MTL neural network in a KD setup. As a consequence, the MTL neural network can be trained on larger datasets as the lack of ground truth is not a limitation any more.

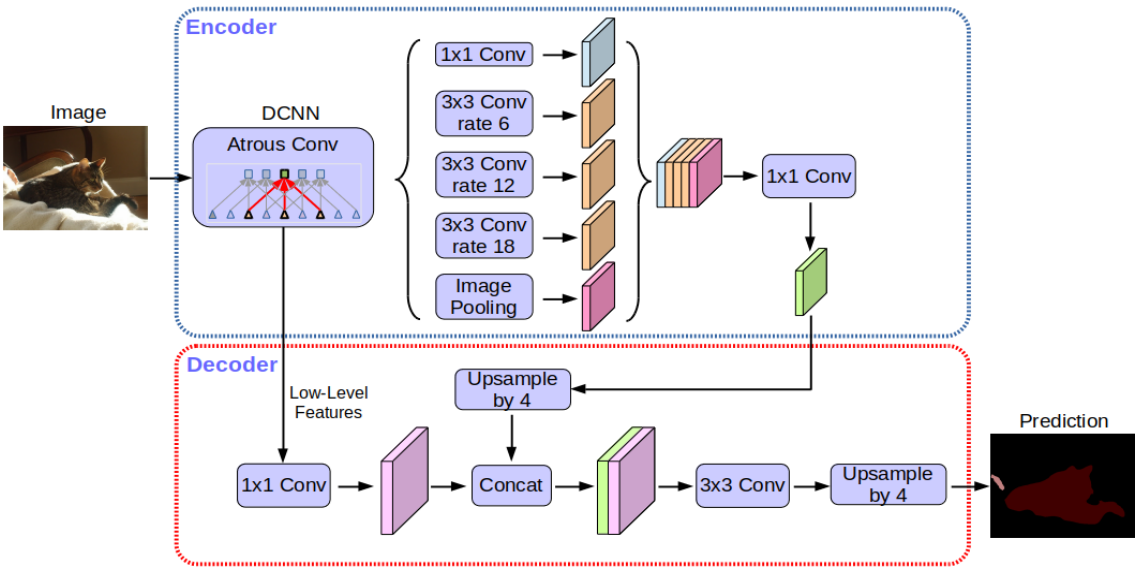
2.2 Semantic segmentation

Semantic segmentation task is a fine-grained class prediction where a label is assigned to each pixel of the image. As the segmentation models need to generate a mask with the same size as the input image, most of the neural networks consist of an encoder which is responsible for the feature extraction and then the decoder that projects the knowledge into the corresponding space. It is important to mention that this kind of task usually lack large datasets with annotations due to the fact that it is an arduous work to produce it.

One possible approach to this challenge is region-based segmentation, where regions of interest are extracted first and then the segmentation task is performed for each of those regions separately. As a final step, the regions are combined to form the final output. For instance, Mask-RCNN [20] uses this methodology to make instance segmentation resulting in state-of-the-art performance.

A second approach is fully convolutional neural networks. This method uses NN employing convolutional and pooling layers so it can accept input with different sizes. This kind of approach can be seen in [17]. However, the direct output of this type of models have low resolution resulting in rather not clear shapes. Due to this problem, deconvolutional layers are introduced to increase the resolution [18].

Another possible approach is weakly supervised segmentation. This option focuses on solving the lack of annotated data using, for instance, other type of annotations such as bounding boxes or contours [21] [40] to train the segmentation model.



3 DeepLab structure [15]

DeepLab [16] is an essential project in the semantic segmentation field. It was designed by Google and went through several changes to improve performance and efficiency. This model uses cutting-edge algorithms such as atrous convolution [22], depthwise convolution [23] and others to obtain state-of-the-art results and lead this challenge during the last years.

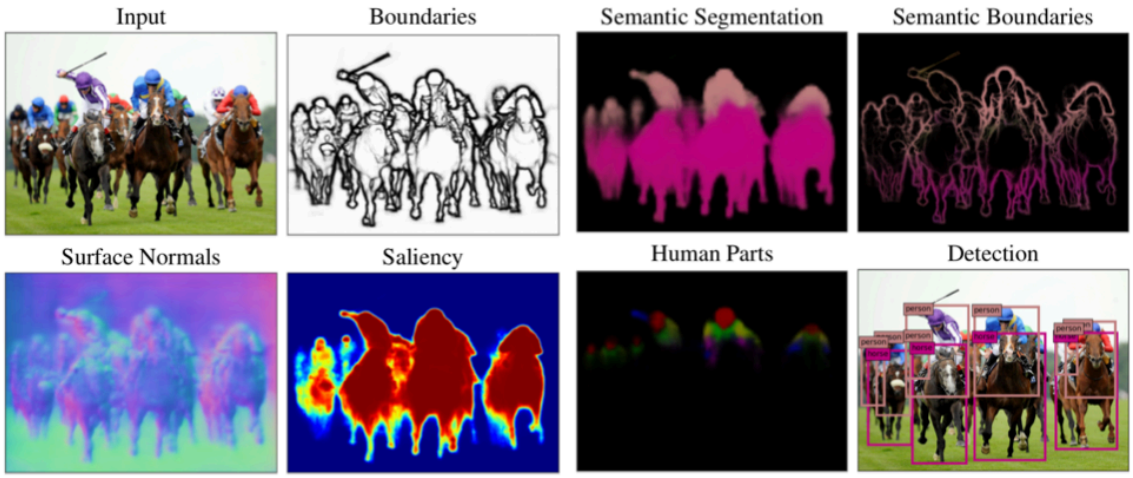
Due to design and time constrictions, on this project, we use Mask-RCNN [20] as the segmentation "tutor" even though it is an instance segmentation NN, instead of using directly a semantic segmentation NN as the projects presented above. The Mask-RCNN [20] output is treated and transformed so it follows the semantic segmentation objective.

2.3 Multi-task learning

Most of the deep learning models nowadays are oriented to solve only one task, such as classification and segmentation, reaching success in their respective field. However, most of these models end up learning to focus on the same features and sharing most of their neural network structure. Moreover, none of them reach a complete understanding of the reality or the image processed, as they are focused on a minor task. In order to achieve full interpretation, different models would have to be used addressing different tasks. But, this approach is clearly computationally inefficient and wouldn't exploit the shared knowledge between tasks. Multi-task learning emerged to give a unified solution to these issues.

MTL refers to the machine learning models that are trained to perform several tasks at the same time, having as objective the optimization of metrics from each task and to generate the corresponding output for each of them. Many projects [29] [30] have shown over the years how MTL is able to generalize better and obtain better performance in some related task by sharing their knowledge representation and aggregating their learning process. Besides, it allows the model to focus on those image features that are important for every task objective.

However, in most of the cases, the datasets used are mostly sparse as they don't have the corresponding ground truth for every task. As a consequence, the MTL models have to change their learning process to adapt to this circumstance, as UberNet [28] does by introducing an asynchronous variant of backpropagation. BlitzNet [27] on the other hand, tries to fix the lack of segmentation ground truth by using an augmentation dataset that contains segmentation contours instead of complete segmentation masks.



4 UberNet output example [28]

There are two main different approaches to design MTL neural networks [26]. Soft parameter sharing defines the neural networks where each task has its own layers but then their parameters are forced to be similar using regularization techniques. On the other hand, hard parameter sharing has hidden layers shared between every task and then, specific-task layers to generate different outputs for each task. This last case is commonly used resulting in projects such as BlitzNet [27] and UberNet [28].

In contrast with the projects presented above, we use the soft output of NN already trained on the corresponding tasks to fill the ground truth missing in the sparse dataset. Consequently, the loss function is modified to completely exploit this kind of data during the training process.

2.4 Knowledge distillation

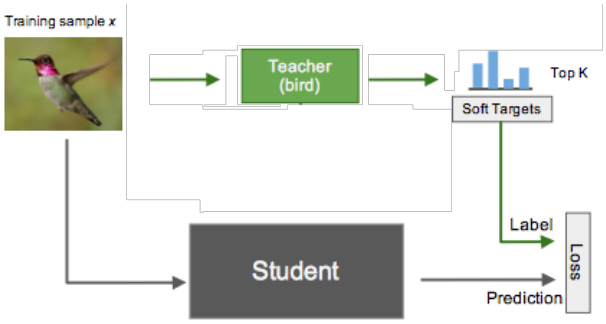
Knowledge distillation is a procedure to train a model using a second neural network as reference instead of learning directly from the data's ground truth. This setup was first introduced in 2015 by Hinton [38] with the purpose of becoming a model compression technique. In this paper is described how the soft output of a NN already trained on a task ("tutor") can be used as an alternative to raw data and its ground truth to train another deep neural network ("student"). Consequently, the "student" NN learns to imitate the professor behavior without the need of the ground truth.

Besides, the paper study how changing the temperature parameter in the SoftMax function can improve how the "student" learns. By defining a high Temperature, the differences between classes values will be less notable than when defining a temperature close to zero. Then, this parameter influences how the "student" learns the similarities and differences between them.

$$SoftMax_i = \frac{e^{(\frac{z_i}{T})}}{\sum_i e^{(\frac{z_j}{T})}}$$

5 SoftMax formula

Other papers [36] [37] continue studying how this technique could be exploited to compress models and introduce several developments such as using intermediate hidden layers and its knowledge representation to help the "student" NN to imitate more accurately the "tutor" behavior.



6 Knowledge distillation example [34]

On the other hand, other projects [35] [39] detected how this technique could be used to further improve model performance instead of only compressing. The experiments executed show how the "student" significantly outperforms the same NN learning directly from the ground truth, even converging faster. Furthermore, the "student" models learn to generalize better and are less likely to overfitting.

On this project, instead of focusing on how KD can be used as a compression technique or to improve one-task NN performance replacing the ground truth with the "tutor" soft output, we try to combine the available ground truth with the "tutors" output to overcome the MTL data limitation and exploit the shared knowledge between tasks in every image.

3 Methodology

In this project, we work with BlitzNet [27] as the multi-task NN, which will play the role of the "student" in the knowledge distillation setup and will be trained on VOC 2012 [42]. As this neural network has two tasks: object detection and segmentation, we use two "tutor" NN that have already been trained on COCO [41] and will generate the soft output needed for BlitzNet [27]. Then, the performance changes generated by the KD setup are studied in different situations and the results are evaluated.

3.1 Data

As described above, the "student" network is trained on VOC 2012 dataset [42] using the corresponding ground truth for object detection and semantic segmentation apart from the tutor's soft output for each of the tasks. In order to truly implement transfer learning between models and simulate the situation where the "tutors" haven't seen the dataset used to train the "student", NN trained with COCO dataset [41] are used as "tutors" for both tasks.

Dataset	Nº images	Nº images without segmentation ground truth	Nº images without detection ground truth
VOC2012seg training set	1464	0	0
VOC2012seg val set	1449	0	0
VOC2012det training set	5717	4566 (80%)	0
VOC2012det trainVal set	11540	9265 (80%)	0
Augmentation dataset (contours)	10582	9118 (86%)	0

7 VOC 2012 [42] datasets

VOC [42] dataset offers different file sets depending on the task the model is going to be trained for. The training and validation sets for segmentation are the only ones which offer both task ground truth for every image. As a consequence, this validation set is used to evaluate every model performance and the training set is used to test the rate of missing ground truth that can be managed by KD.

The training and trainval sets for detection are significantly larger than segmentation's. However, they miss segmentation GT for 80% of the images. Using the "tutor" soft output, we intend to show that even though these datasets lack the GT needed, they can still be used to train a multi-task neural network. Nevertheless, the trainval set for the detection task shares files with the validation set used in this project, so even though it would have been representative to use a dataset almost ten times larger than segmentation training set, it couldn't be used for this purpose.

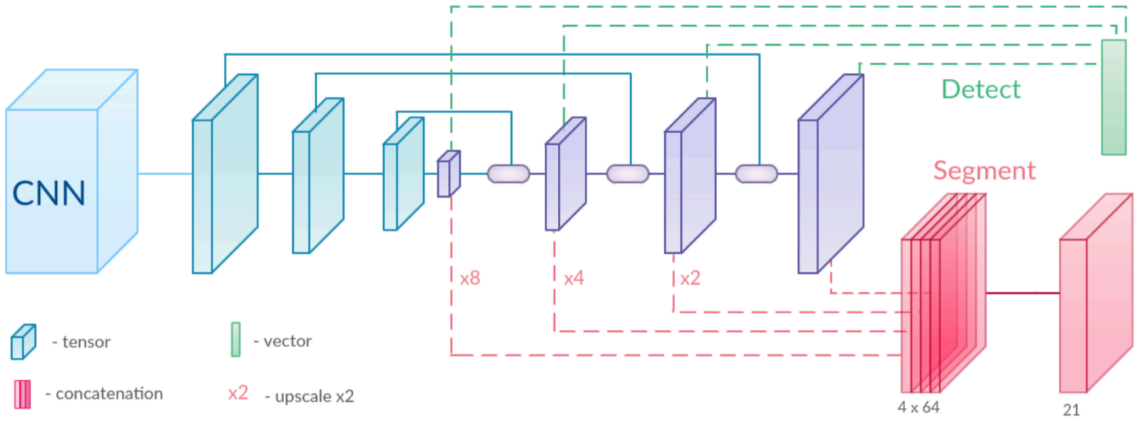
Furthermore, as it is explained in the BlitzNet [27] paper, an augmentation dataset [40] is used to improve the performance of the models as the number of VOC images which have ground truth for both tasks is limited. This dataset contains semantic contours instead of the complete semantic segmentation data needed, but it has been proven to be helpful for semantic segmentation.

3.2 BlitzNet

BlitzNet [27] is a fully convolutional neural network that performs object detection and semantic segmentation. Its architecture has a ResNet-50 convolutional neural network, pretrained on ImageNet [44] for image recognition [11], that is used as the encoder to extract the image features.

Imitating the SSD structure [9], the encoder is followed by a set of convolutional and pooling layers to obtain a sequence of feature maps with gradually decreasing spatial resolution and increasing field of view so a multi-scale search of bounding boxes is performed.

Then, following the approach from [17] and [18], the feature maps are up-scaled with deconvolutional layers in order to generate precise segmentation masks. Along this upscale stream, skip connections [43] are used to combine feature maps from the downscale and upscale streams.



8 BlitzNet architecture [27]

With one last convolutional layer for each of the tasks, the output is produced by predicting a class for each bounding box from the upscale stream feature maps and, separately, forecast the pixel labels by upscaling the activations of the upscale stream, concatenating them and feeding it to the final layer.

The loss function used is the sum of each task loss. For segmentation, the loss is the cross-entropy between predicted and ground truth label distribution. On the other hand, the bounding boxes from the detection task are transformed to match the anchor boxes format generated by BlitzNet [27] and then the loss function from SSD [9] is used.

3.3 Tutors

For the object detection task, Faster R-CNN [7] was used as the "tutor" in the KD setup. As it is explained in section 2.1, Faster R-CNN [7] has been on the lead of the object detection field during the last years (its architecture is shown in 2.1 figure). As we needed a "tutor" trained on a different dataset than VOC [42], we used the model from [12] trained on COCO [41], using as base network the ResNet-152 which showed the best performance on this dataset.

In order to decide if it was appropriate to use this model as BlitzNet [27] detection "tutor", we evaluated its performance on VOC 2012 test set, previously discarding all those predicted boxes with classes that didn't belong to the VOC labels. It showed a 0.7979 mAP (mean average precision), a bit higher than BlitzNet512 trained on VOC (0.79 mAP) and only overcome by BlitzNet [27] models trained on both COCO [41] and VOC [42] datasets as shown in the paper (0.838 mAP). Once this comparison was made, we decided to use it as the detection "tutor" in our project.

On the other hand, for the segmentation task it was impossible to find a DeepLab [16] model trained only on COCO [41] and due to time constrains, it was decided to not train one from scratch. First, a model trained on COCO-Stuff [19] was tested on the VOC2012seg val set but its performance only reached 0.23 mIoU (mean intersection over union), not being comparable to 0.75 mIoU, the performance of BlitzNet [27] trained on VOC [42].

Ultimately, Mask R-CNN [20] was tested on VOC 2012 [42] val set. This model is an extension of Faster R-CNN [7], with ResNet-101 as backbone neural network, used for instance segmentation. Apart from generating a bounding box offset and its label as output, it adds a new parallel branch with a series of convolutional layers to output a binary mask for each bounding box. Each binary mask only has labels for background and the corresponding class predicted for the RoI.

In order to evaluate Mask R-CNN [20] on VOC2012seg [42], we discard the bounding boxes and the associated binary masks that don't belong to VOC [42] classes. Then, a VOC [42] labels vector is created for each pixel containing a confidence value on a specific label position in case a bounding box corresponding to that class surrounds that pixel and the corresponding binary mask has a positive value in that same pixel. The value from the generated VOC labels vector corresponds to the confidence value assigned to the bounding box produced by Mask R-CNN [20].

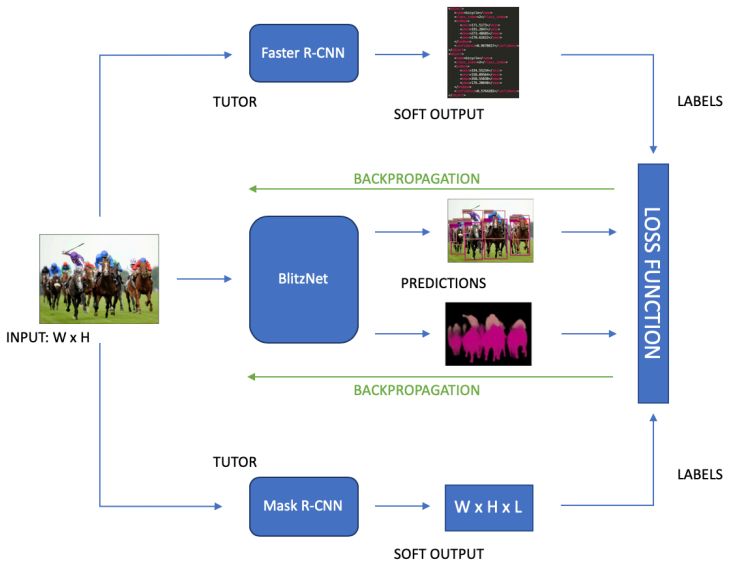
Model	KD role	Training dataset	Test dataset	Tasks
Mask R-CNN [20]	Tutor	COCO [41]	VOC12 [42]	Segmentation
Faster R-CNN [7]	Tutor	COCO [41]	VOC12 [42]	Object detection
BlitzNet [27]	Student	VOC12 [42]	VOC12 [42]	Object detection and segmentation

9 Models, data and tasks

As a direct consequence of this mechanism, most of the pixel vectors are sparse, only having more than one value those pixels with several bounding boxes (from different classes) around them and with the corresponding masks having a positive value in more than one case.

The results showed that its performance, 0.756 mIoU, was similar to BlitzNet [27] trained with the augmented dataset [40] for VOC 2012 [42] (0.757 mIoU). Therefore, Mask R-CNN [20] was used as the segmentation "tutor" in the KD set up.

3.4 Knowledge distillation

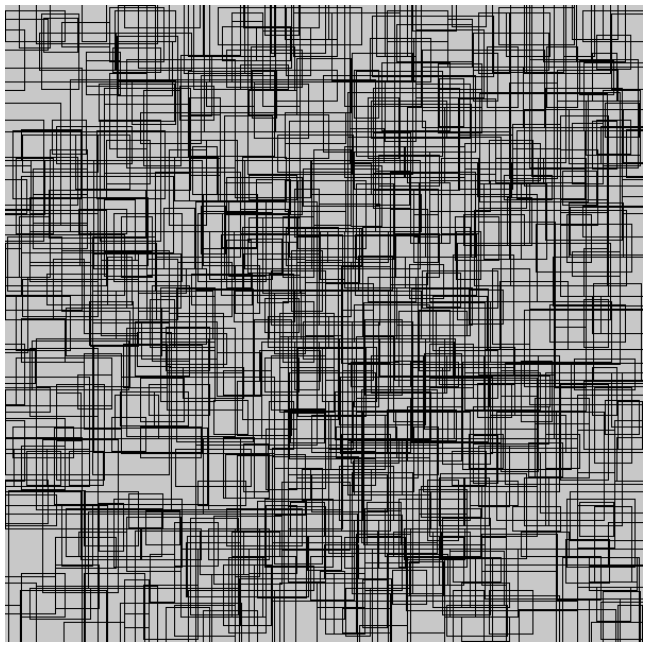


10 Knowledge distillation pipeline on BlitzNet [27]

For the sake of training speed, we decided to not implement the complete learning pipeline using "tutors" and "student" NN. Instead, the process was made in two steps. First, the "tutors" went through the "student" training set and generated the corresponding soft output for each of the tasks. These outputs were saved and would be read directly by the "student" NN during training instead of reading the ground truth values. In the case of the detection task, the "tutor" generates a set of bounding boxes with its location, its predicted class and the confidence. The segmentation "tutor" on the other hand, for each image, saves a three dimensions array (height x width x labels) obtained from the pixel vectors.

In order to learn from the "tutors" soft output, the loss function had to be modified. In the case of the detection, the results from the cross-entropy function used in BlitzNet [27] are multiplied by the confidence of the corresponding bounding box generated by the detection "tutor". At this point is important to be aware that in order to learn from the ground truth, detection models transform the ground truth boxes to match the RoI grid used to make the predictions. Each box location is matched to those RoI which

have an intersection superior to a threshold, generating an offset location and propagating the class label accordingly. In order to use the confidence values in the loss function, we propagated these values following the same criteria as the class labels.



11 RoI grid used for object detection training [1]

However, detection models also learn from those predictions which don't have any ground truth match and, following this confidence propagation algorithm, those examples wouldn't have any confidence assigned. Several experiments were executed to clarify if the confidence values of the "tutor" predictions could be used in the KD set up and how it would affect the "student" performance (section 4.2).

The second problem was related to the number of bounding boxes generated by the "tutor" per image. Each image was assigned an average of 100 different bounding boxes, most of them having a confidence value lower than 0.1%. As these bounding boxes with low confidence could confuse the "student", it was decided to check if filtering out those bounding boxes could improve the "student" performance.

For the segmentation loss, we had to adjust it so instead of learning from a single label per pixel, BlitzNet [27] could learn from the label vector that each pixel had assigned in the soft output from the segmentation "tutor". In order to do so, a SoftMax function is first applied to the "tutor" output, evaluating how different values of the SoftMax temperature would modify the "student" performance. Once the SoftMax is applied, the cross-entropy function is used between the logits of the BlitzNet [27] segmentation prediction and the normalized soft output of the "tutor".

3.5 Evaluation method

To evaluate the different models, we use the VOC2012seg validation dataset, as it is the only validation set that contains both tasks ground truth for every image. The evaluation measures used are mean intersection over union (mIoU) for the segmentation task and mean average precision (mAP) for detection.

4 Experiments

In order to check how the KD technique, with Mask R-CNN [20] and Faster R-CNN [7] as "tutors", could help a MTL neural network, such as BlitzNet [27], to be trained on several tasks, we followed the next steps:

- As the first step, we train BlitzNet [27] on VOC2012seg [42] dataset without the participation of any "tutor" in order to obtain a baseline to compare with. Then, for each of the tasks, we run one experiment substituting every image ground truth with the corresponding soft output of the "tutor".

- Secondly, the "tutors" output is calibrated to exploit the knowledge transferred to the "student" (section 3.4).

- Then, several experiments are executed using the VOC2012seg [42] training set to simulate different percentages of missing ground truth, so it is shown how helpful the "tutors" are depending on the number of images using their output.

- Afterward, VOC2012det [42] dataset, which misses 80% of the segmentation GT but is almost five times larger than the segmentation dataset, is used to show the advantage of having a KD setup covering the missing data of a large dataset.

- Subsequently, the ground truth and "tutors" output are combined on every image loss function instead of only using the "tutors" whenever the image misses the corresponding task ground truth.

4.1 Initial knowledge distillation experiments

With the intention of knowing if a KD set up could help MTL neural networks to overcome the lack of ground truth for a specific task, we run three different experiments training the models on VOC2012seg [42] training set (which doesn't miss ground truth for any of the two tasks) and evaluate them on the corresponding val dataset:

- BlitzNet [27] is trained using the ground truth for both tasks. This NN will show the performance the other models should reach, as it would mean that BlitzNet [27] can obtain its optimum performance and overcome the lack of GT making use of a KD technique.

- BlitzNet [27] is trained using the ground truth for the objection detection task, but the semantic segmentation GT is removed and replaced with the soft output of Mask R-CNN [20]. In this experiment, the temperature used in the "tutor" SoftMax function was equal to 0,4.

- BlitzNet [27] is trained using the ground truth for semantic segmentation, but the object detection GT is substituted with the Faster R-CNN [7] output. The "tutor" predictions were first filtered using a confidence threshold equal to 0.001. Then, its prediction confidences were used in the BlitzNet [27] loss function as explained in section 3.4

Model	Segmentation tutor	Detection tutor	mAP (det)	mIoU (seg)
BlitzNet [27]	None	None	0.626	0.602
BlitzNet [27]	Mask R-CNN [20]	None	0.624	0.523
BlitzNet [27]	None	Faster R-CNN [7]	0.640	0.634

12 Comparison of detection and segmentation performance on VOC2012seg val dataset (models were trained on VOC2012seg training set)

The results show that in the case of replacing the object detection ground truth with the Faster R-CNN [7] soft out, BlitzNet [27] not only presents similar performance to the model trained without "tutors", it even exposes slightly better performance on both tasks (table 12). On the other hand, the model trained with the Mask R-CNN [20] output couldn't reach similar performance to the BlitzNet [27] original model (trained without "tutors"), obtaining an inferior performance on the segmentation task and similar on the detection, as this last task is not missing GT in the experiment and is not so affected by the segmentation "tutor" (table 12).

4.2 Tutors calibration

As it has been explained in section 3.3, both "tutors" output could be used by BlitzNet [27] in different ways, resulting in different performance. In order to stablish the best configuration to exploit the KD setup, we run several experiments on the VOC 2012 [42] datasets.

On the object detection side, it was needed to check if the confidence values predicted by Faster R-CNN [7] for each bounding box could be used in the "student" loss function and improve its performance. Additionally, it was important to verify if using "tutor" low confidence predictions could confuse the "student" and try to define a threshold so the problem could be avoided. Therefore, the following experiments were executed:

- BlitzNet [27] is trained using Faster R-CNN [7] soft output without filtering out any predicted bounding box and having the "tutor" prediction confidences included in the loss function.

- BlitzNet [27] is trained with the Faster R-CNN [7] output but filtering out first the predicted bounding boxes with low confidence. In order to define the appropriate threshold, separately, we generated the Faster R-CNN [7] output for VOC2012det [42] test set and filtered it following different threshold values. Then, we selected the threshold that led to the best model performance, 0.001 in

this case. In this experiment, the prediction confidences are used in the BlitzNet [27] loss function.

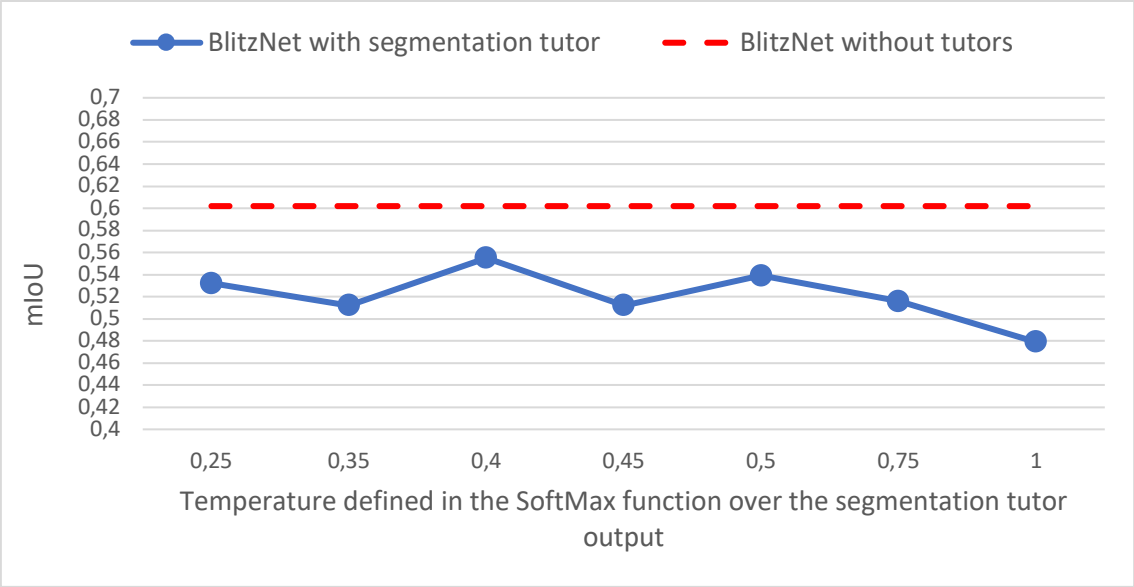
- BlitzNet [27] is trained again with the Faster R-CNN [7] soft output filtered with the confidence threshold equal to 0.001 but in this case the prediction confidences are not used in the BlitzNet [27] loss function.

Model	**Segmentation tutor**	**Detection tutor**	**Filtering low confidence tutor predictions (threshold 0.001)**	**Confidences from tutor predictions used in student loss**	**mAP (det)**	**mIoU (seg)**
BlitzNet [27]	None	Faster R-CNN [7]	No	Yes	0.610	0.599
BlitzNet [27]	None	Faster R-CNN [7]	Yes	Yes	0.640	0.634
BlitzNet [27]	None	Faster R-CNN [7]	Yes	No	0.414	0.543
BlitzNet [27]	None	None	-	-	0.626	0.602

13 BlitzNet performance depending on the detection tutor configuration (models were trained on VOC2012seg training dataset and tested on the val dataset)

The experiments show that when filtering out the Faster R-CNN [7] predictions with low confidence, the "student" model converges faster than when including those low confidence boxes in the training data. Besides, its performance is actually better on both tasks than BlitzNet [27] trained on the same dataset without "tutors" (table 13). It reveals as well that defining a loss function using the confidence values from the "tutor" predictions, makes a meaningful difference in the "student" final performance on both tasks (table 13).

In the case of the segmentation "tutor", Mask R-CNN [20], several experiments were executed to define the temperature value in the SoftMax formula that would exploit better the knowledge transferred to BlitzNet [27]. In figure 14, we show how BlitzNet [27] performance changes as the temperature value from the Mask R-CNN [20] SoftMax function is modified.



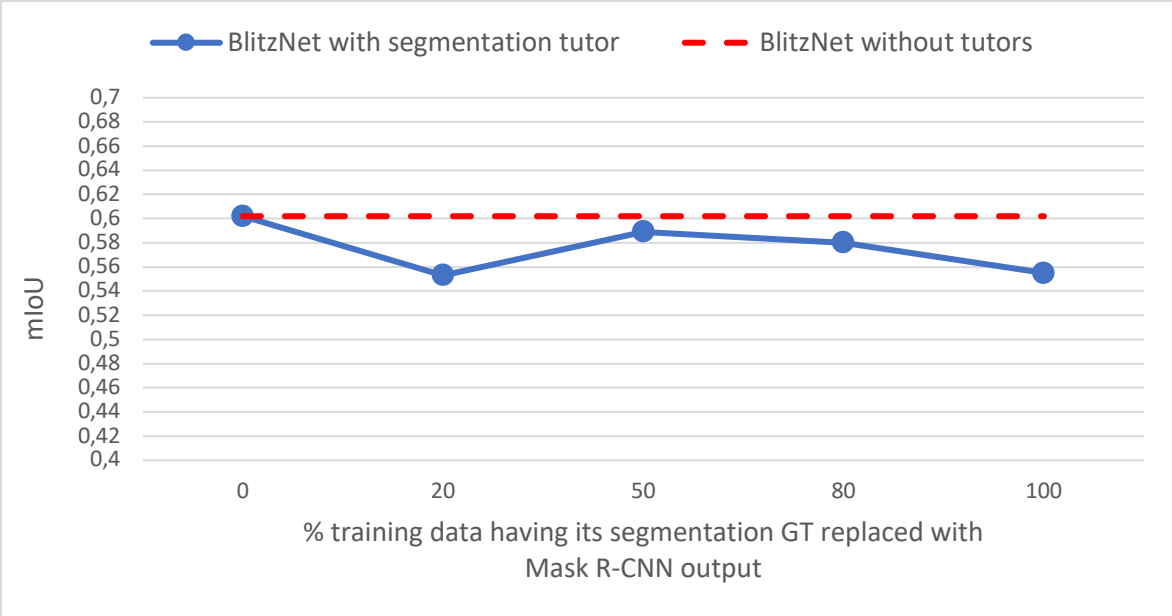
14 BlitzNet performance depending on the segmentation tutor configuration (models were trained on VOC2012seg training dataset and tested on the val dataset)

The plot shows that a temperature value equal to 0,4 caused BlitzNet [27] to have the best performance on the semantic segmentation task, 0,555 mIoU. However, none of these experiments generated a BlitzNet [27] model that outperforms the same model trained with the complete dataset ground truth, 0,602 mIoU.

4.3 Different rate of ground truth missing on the dataset

As it is shown in table 1 and 7, datasets usually lack a specific task GT for a subset of training images, however, the previous experiments simulate the lack of GT for 100% of the training data. At this stage, the objective of the experiments was to simulate the lack of ground truth for a specific task in a training data subset and check how helpful the KD "tutors" would be depending on the size of the subset. In order to do so, we worked with VOC2012seg [42] dataset as it is the only dataset that contains both task GT for every image and therefore, we wouldn't have any problem defining the subsets.

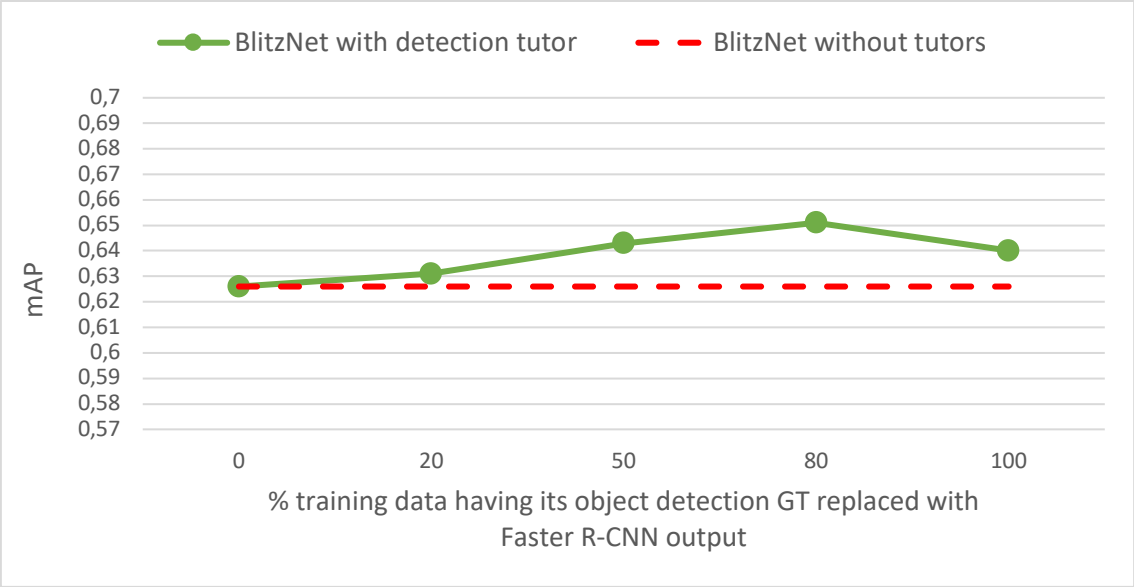
For this purpose, we first oriented the experiments to the semantic segmentation task, leaving the ground truth for object detection accessible during training, and we defined different subset sizes for which we would substitute the segmentation GT with the Mask R-CNN [20] soft output (using a temperature value equal to 0,4). Figure 15 shows how the BlitzNet [27] mIoU changes as the Mask R-CNN [20] subset size is modified.



15 BlitzNet performance depending on the number of images having its segmentation GT replaced with Mask R-CNN [20] output (models were trained on VOC2012seg training dataset and tested on the val dataset)

The results show that BlitzNet [27] semantic segmentation performance is better when the subset is none existent, in other words, when no image is missing GT and the "tutor" is not needed. From the other cases, the model exhibits the best performance (0,589 mIoU) when half of the training dataset is missing GT. Besides, it was expected that the "student" performance would be close to BlitzNet [27] trained without "tutors" as the number of images using Mask R-CNN [20] output decreased, however, figure 15 shows how this trend is not followed.

On the other hand, we run similar experiments for the object detection task, leaving on this case the semantic segmentation GT accessible during training and using the Faster R-CNN [7] output for detection (filtering low confidence predictions and using the confidence vector in the BlitzNet [27] loss function). Independent from the size of the Faster R-CNN [7] subset, all the experiments show better performance than the BlitzNet [27] trained without "tutors". Figure 16 shows how BlitzNet [27] performance improves as the number of images using Faster R-CNN [7] output increases. However, the best performance, 0.651 mAP, was reached when the subset of images missing GT was 80% of the training set, not 100%.



16 BlitzNet performance depending on the number of images having its detection GT replaced with Faster R-CNN [7] output (models were trained on VOC2012seg training dataset and tested on the val dataset)

4.4 Using a larger dataset using tutors' soft output

MTL models are usually trained on small datasets due to the fact that most of the large ones don't have the required GT for every task on every image. Taking advantage of the fact that the KD "tutors" can generate GT for any data point, we run experiments with a larger dataset. In all previous experiments, the models were trained on VOC2012seg dataset (1464 images) as it is the only one with GT for segmentation and object detection on every image, but in this section, we train BlitzNet [27] on VOC2012det (5717 images) filling the missing GT for the segmentation task with the corresponding output of Mask R-CNN [20] (using a temperature equal to 0,4). In this experiment, Faster R-CNN [7] is not used because no image is missing object detection GT.

Model	Segmentation tutor	Detection tutor	mAP (det)	mIoU (seg)
BlitzNet [27]	None	None	0.726	0.646
BlitzNet [27]	Mask R-CNN [20]	None	0.702	0.644

17 Comparison of detection and segmentation performance on VOC2012seg val dataset (models were trained on VOC2012det training set)

Contrary to the expectations, BlitzNet [27] trained without "tutors" shows better performance in both tasks than the same model trained using Mask R-CNN [20] output to fill the lack of segmentation GT even though the training dataset for both experiments misses 80% of GT (table 17). This could be caused by the fact that Mask R-CNN [20] (semantic segmentation "tutor") generates a sparse output due to its design.

As a consequence, the "student" NN, BlitzNet [27] in our case, is not able to completely exploit the advantages of KD and it even undermines its performance.

4.5 Ground truth and tutor losses combined

In the final experiment, the objective was to make use of both "tutors" output on every training image and check if the "student" performance would suffer meaningful improvements. In all previous experiments, the loss function for the "tutor" output would only be used on those images missing GT. On the contrary, during the BlitzNet [27] training process on these experiments, the loss function was the weighted sum of the GT and "tutors" loss on every image, making use of the Faster R-CNN [7] output (filtering low confidence predictions and using the confidence vector in the BlitzNet [27] loss function) for detection and the Mask R-CNN [20] output (using a temperature value equal to 0,4) for segmentation.

$$L_{DET} = 0.5 * L_{GT_{DET}} + 0.5 * L_{TUTOR_{FASTER\ R-CNN}}$$
$$L_{SEG} = 0.5 * L_{GT_{SEG}} + 0.5 * L_{TUTOR_{MASK\ R-CNN}}$$
$$L_{COMB} = L_{DET} + L_{SEG}$$

18 Sum of GT and tutors loss

BlitzNet [27] using L_{COMB} loss function shows better object detection performance than BlitzNet [27] trained without "tutors", regardless of the training set size. However, the difference is not as big as in the experiments from section 4.1. This could be caused by the influence of Mask R-CNN [20] output, that even though it is the segmentation "tutor", has an impact on the object detection performance of BlitzNet [27] as revealed in experiments from sections 4.1 and 4.4.

Besides, the segmentation performance of BlitzNet [27] using L_{COMB} loss function is worse than of BlitzNet [27] trained without "tutors", following the same trend we have been observing in all previous experiments that made use of Mask R-CNN [20] output.

Model	Segmentation tutor	Detection tutor	Training dataset	mAP (det)	mIoU (seg)
BlitzNet [27] with L_{COMB}	Mask R-CNN [20]	Faster R-CNN [7]	VOC2012seg	0.636	0.55
BlitzNet [27]	None	None	VOC2012seg	0.626	0.602
BlitzNet [27] with L_{COMB}	Mask R-CNN [20]	Faster R-CNN [7]	VOC2012det	0.736	0.629
BlitzNet [27]	None	None	VOC2012det	0.726	0.646

19 Comparison of detection and segmentation performance on VOC2012seg val dataset

5 Discussion and future work

The master thesis proved to be a complicated project presenting many challenges from the beginning. The first problem was the need to find the right "tutor" for each of the task and as a consequence, requiring to get familiar with each of the projects with its own tools and platforms. Besides, in each of them, changes needed to be introduced in their design in order to meet BlitzNet [27] needs, such as having a specific output structure so it was possible to learn from it. Another important issue was the need to modify BlitzNet [27] loss function and the data loading process with the purpose of adapting to the different kinds of data. Moreover, the main project limitation was the amount of time and resources that BlitzNet [27] needed to be trained due to its size and design.

The experiments executed introducing knowledge distillation in a MTL neural network present quite different results depending on the task and "tutor" used. In the case of object detection, BlitzNet [27] models which were trained with Faster R-CNN [7] soft output replacing the corresponding GT, showed meaningful improvements in its performance (figure 16). The fact that, for each training image, Faster R-CNN [7] offers a larger number of bounding boxes than the GT, and a more complete information involving prediction confidences, results in BlitzNet [27] models, trained with this data, having a better understanding and better performance in both tasks than a BlitzNet [27] model trained without any "tutor" output (table 12).

These experiments suggest that the KD technique and the right "tutor" could help MTL neural networks to not be undermined by the lack of GT in large datasets and even further improve their performance with a more complete training data than the default ground truth. However, these examples are working in the situation of lacking GT for the object detection task, and this is not a common case as it is shown in tables 1 and 7.

On the segmentation "tutor" side, the models generated when Mask R-CNN [20] output was used to train BlitzNet [27] are not that satisfactory. In figures 14 and 15, the results show how using the Mask R-CNN [20] soft output only weakens the model instead of offering advantages. Even when training on larger datasets and filling the missing GT with the "tutor" output, the model kept showing worse performance than training the same model without "tutor" on the same dataset, even though the dataset lacks 80% of the segmentation GT (table 17).

This outcome could be caused by the structure of the Mask R-CNN [20] output. Due to its design (explained in section 3.3), Mask R-CNN [20] generates a sequence of binary masks instead of a label's vector with probabilities. As a consequence, its final output is extremely sparse and BlitzNet [27] is not capable of extracting profound knowledge out of it, in contrast with the Faster R-CNN [7] output which is clearly more complete than the GT. Due to the fact that the segmentation "tutor" needed to be trained on a dataset different than the "student" (explained in section 1.2) and the time constraints, we couldn't find a better semantic segmentation model to act as KD "tutor" in the project or either train a different one from scratch.

As a conclusion, we think that reproducing the same experiments with another semantic segmentation "tutor" could be attempted, making sure that its output offers a more complete and less sparse data than Mask R-CNN [20]. Once satisfactory results have been achieved in both tasks, the idea of using KD in MTL could be further developed in many different ways. One possibility could be to include new tasks, such as instance segmentation, in BlitzNet [27], making the corresponding changes in its structure, and study if KD would still be helpful when the number of tasks increases. Another line of work could be to introduce the same KD technique in a new MTL neural networks, such as UberNet [28], and research if the results keep showing improvements regardless of the "student" neural network used. A third research could be done over the idea of combining GT and "tutors" output following new approaches and study the changes in the "student" neural network performance. Finally, once KD technique has proven to be a key instrument in MTL, the objective could be finding the optimum "tutor" for each task and each MTL neural network and extend the datasets that lack the corresponding GT so MTL researches don't face any data limitation anymore and the topic can be studied from different perspectives.

The project is available at:
https://github.com/AlejandroHernandezMunuera/BlitzNet_Knowledge_Distillation

5.1 Acknowledgements

Thanks to TUB and the Neural Information Processing department for its support and resources offered to carry out this project. I would particularly like to thank Youssef Kashef for his exceptional help and patience.

Acronyms

MTL	Multi-Task Learning
NN	Neural Network
KD	Knowledge Distillation
GT	Ground Truth
RoI	Region of Interest

Bibliography

[1] Anders Christiansen. Anchor boxes – The key to quality object detection. October, 2018. Medium. https://medium.com/@andersasac/anchor-boxes-the-key-to-quality-object-detection-ddf9d612d4f9.

[2] Building a production grade object detection system with SKIL and YOLO. January, 2019. Skymind. https://blog.skymind.ai/building-a-production-grade-object-detection-system-with-skil-and-yolo/.

[3] Jonathan Hui. What do we learn from region based object detectors (Faster R-CNN, R-FCN, FPN)?. March, 2018. Medium. https://medium.com/@jonathan_hui/what-do-we-learn-from-region-based-object-detectors-faster-r-cnn-r-fcn-fpn-7e354377a7c9.

[4] Rohith Gandhi. R-CNN, Fast R-CNN, Faster R-CNN, YOLO – Object detection algorithms. July, 2018. Medium. https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e.

[5] Ross Girshick, Jeff Donahue, Trevor Darrel, Jitendra Malik. UC Berkeley. Rich feature hierarchies for accurate object detection and semantic segmentation. 2014. arXiv: 1311.2524.

[6] Ross Girschick. Microsoft Research. Fast R-CNN. 2015. arXiv: 1504.08083.

[7] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. 2016. arXiv: 1506.01497.

[8] Pierre Sermanet, David Figen, Xiang Zhang, Michael Mathieu, Rob Fergus, Yann LeCun. Courant Institute of Mathematical Science, New York University. OverFeat: Integrated recognition, localization and detection using convolutional networks. 2014. arXiv: 1312.6229.

[9] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg. UNC Chapel Hill, Zoox Inc, Google Inc, University of Michigan. SSD: Single Shot multibox Detector. 2016. arXiv: 1512.02325.

[10] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Fathadi. University of Washington, Allen Institute for AI, Facebook AI Research. You Only Look Once: Unified, real-time object detection. 2016. arXiv: 1506.02640.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Microsoft Research. Deep residual learning for image recognition. 2015. arXiv: 1512.03385.

[12] Xinlei Chen. Tf-faster-rcnn GitHub repository.

[13] James Le. How to do semantic segmentation using deep learning. May, 2018. Medium. https://medium.com/nanonets/how-to-do-image-segmentation-using-deep-learning-c673cc5862ef.

[14] Saurabh Pal. Semantic segmentation: Introduction to the deep learning technique behind google pixel's camera. February, 2019. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2019/02/tutorial-semantic-segmentation-google-deeplab/.

[15] Liang-Chieh Chen, Yukun Zhu. Semantic image segmentation with DeepLab in TensorFlow. March, 2018. Google AI Blog. https://ai.googleblog.com/2018/03/semantic-image-segmentation-with.html.

[16] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, Alan L. Yuille. DeepLab: Semantic Image Segmentation with deep convolutional nets, atrous convolution and fully connected CRFs. 2017. arXiv: 1606.00915.

[17] Evan Shelhamer, Jonathan Long, Trevor Darrel. Fully convolutional networks for semantic segmentation. 2016. arXiv: 1605.06211.

[18] Hyeonwoo Noh, Seunghoon Hong, Bohyung Han. Department of Computer Science and Engineering, POSTECH, Korea. Learning deconvolution network for semantic segmentation. 2015. arXiv: 1505.04366.

[19] Holger Caesar, Jasper Uijlings, Vittorio Ferrari. University of Edinburgh, Google AI Perception. COCO-Stuff: Thing and Stuff classes in context. 2018. arXiv: 1612.03716.

[20] Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick. Facebook AI Research. Mask R-CNN. 2018. arXiv: 1703.06870.

[21] Anna Khoreva, Rodrigo Benenson, Jan Hosang, Matthias Hein, Bernt Schiele. Max Planck Institute for Informatics, Saarland University. Simple does it: Weakly supervised instance and semantic segmentation. 2016. arXiv: 1603.07485.

[22] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, Hartwig Adam. Google Inc. Encoder-Decoder with atrous separable convolution for semantic image segmentation. 2018. arXiv: 1802.02611.

[23] Francois Chollet. Google Inc. Xception: Deep learning with depthwise separable convolutions. 2017. arXiv: 1610.02357.

[24] Matterport Inc. Mask_RCNN GitHub repository.

[25] Kajal Gupta. Multi-task learning with deep neural networks. July, 2017. Medium. https://medium.com/@kajalgupta/multi-task-learning-with-deep-neural-networks-7544f8b7b4e3.

[26] Sebastian Ruder. An overview of multi-task learning in deep neural networks. 2017. arXiv: 1706.05098.

[27] Nikita Dvornik, Konstantin Shmelkov, Julien Mairal, Cordelia Schmid. INRIA. BlitzNet: A real-time deep network for scene understanding. 2017. arXiv: 1708.02813.

[28] Iasonas Kokkinos. INRIA. UberNet: Training a 'Universal' convolutional neural network for low-, mid- and high-level vision using diverse datasets and limited memory. 2016. arXiv: 1609.02132.

[29] Jian Yao, Sanja Fidler, Raquel Urtasun. Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. CVPR, 2012.

[30] Marvin Teichmann, Michael Weber, Marius Zöllner, Roberto Cipolla, Raquel Urtasun. MultiNet: Real-time joint semantic reasoning for autonomous driving. 2018. arXiv: 1612.07695

[31] Sanja Fidler, Roozbeh Mottaghi, Alan Yiller, Raquel Urtasun. Bottom-up segmentation for top-down detection. CVPR, 2013.

[32] Stephen Gould, Tianshi Gao, Daphne Koller. Region-based segmentation and object detection.

[33] Nikita Dvornik. Blitznet GitHub repository.

[34] Ujjwal Upadhyay. Knowledge distillation. April, 2018. Medium. https://medium.com/neural-machines/knowledge-distillation-dc241d7c2322.

[35] Junho Yim, Donggyu Joo, Jihoon Bae, Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. CVPR, 2017.

[36] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, Yoshua Bengio. FitNets: Hints for thin deep nets. 2015. arXiv: 1412.6550.

[37] Chenglin Yang, Lingxi Xie, Siyuan Qiao, Alan Yuille. The Johns Hopkins University. Training deep neural networks in generations: A more tolerant teacher educates better students. 2018. arXiv: 1805.05551.

[38] Geoffrey Hinton, Oriol Vinyals, Jeff Dean. Distilling the knowledge in a neural network. 2015. arXiv: 1503.02531.

[39] Hessam Bagherinezhad, Maxwell Horton, Mohammad Rastegari, Ali Farhadi. Label refinery: Improving ImageNet classification through label progression. 2018. arXiv: 1805.02641.

[40] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, Jitendra Malik. Semantic contours from inverse detectors. ICCV, 2011.

[41] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, Piotr Dollár. Microsoft COCO: Common objects in context. 2015. arXiv: 1405.0312.

[42] Mark Ereringham, Luc Van Gool, Christopher K.I. Williams, John Winn, Andrew Zisserman. The PASCAL visual object classes (VOC) challenge. 2009.

[43] Alejandro Newell, Kaiyu Yang, Jia Deng. Stacked hourglass networks for human pose estimation. 2016. arXiv: 1603.06937.

[44] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, Li Fei-Fei.Princeton University, USA. ImageNet: A large-scale hierarchical image database. CVPR, 2009.