Neural Networks 170 (2024) 254–265

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Multi-Adaptive Optimization for multi-task learning with deep neural networks

Álvaro S. Hervella *, José Rouco, Jorge Novo, Marcos Ortega

Centro de Investigación CITIC, Universidade da Coruña, A Coruña, Spain
VARPA Research Group, Instituto de Investigación Biomédica de A Coruña (INIBIC), Universidade da Coruña, A Coruña, Spain

ARTICLE INFO

Keywords:
Multi-task learning
Computer vision
Neural networks
Deep learning
Optimization
Gradient descent

ABSTRACT

Multi-task learning is a promising paradigm to leverage task interrelations during the training of deep neural networks. A key challenge in the training of multi-task networks is to adequately balance the complementary supervisory signals of multiple tasks. In that regard, although several task-balancing approaches have been proposed, they are usually limited by the use of per-task weighting schemes and do not completely address the uneven contribution of the different tasks to the network training. In contrast to classical approaches, we propose a novel Multi-Adaptive Optimization (MAO) strategy that dynamically adjusts the contribution of each task to the training of each individual parameter in the network. This automatically produces a balanced learning across tasks and across parameters, throughout the whole training and for any number of tasks. To validate our proposal, we perform comparative experiments on real-world datasets for computer vision, considering different experimental settings. These experiments allow us to analyze the performance obtained in several multi-task scenarios along with the learning balance across tasks, network layers and training steps. The results demonstrate that MAO outperforms previous task-balancing alternatives. Additionally, the performed analyses provide insights that allow us to comprehend the advantages of this novel approach for multi-task learning.

1. Introduction

In recent years, deep learning has become the predominant technique for solving numerous computer vision problems. In this context, Deep Neural Networks (DNNs) are typically trained to solve single specific tasks, such as, e.g., image segmentation (Minaee et al., 2021), object detection (Zhao, Zheng, Xu, & Wu, 2019), or image classification (Hervella, Rouco, Novo, & Ortega, 2021). However, in real-world applications, it is common to face several tasks of interest that share input domain and are related to each other (Almalioglu et al., 2022; Eigen & Fergus, 2015). While using task-specific networks usually provides satisfactory performance, in this context it is also possible to take advantage of the task interrelations by adopting a Multi-Task Learning (MTL) approach (Caruana, 1997). The MTL paradigm leverages the availability of complementary supervisory signals for training models that are capable of simultaneously solving several related tasks. In the context of deep learning, this produces richer representations that integrate complementary aspects of the data not available from a single task. This leads to several advantages over single-task learning. First, the additional multi-task feedback can potentially improve the performance of the different tasks or, equivalently, allow a similar performance with less training samples (Misra, Shrivastava, Gupta, & Hebert, 2016; Standley et al., 2020). Conveniently, this can also be taken advantage of by using unsupervised or self-supervised auxiliary tasks aiming to improve the performance of the main target task. Second, the use of shared representations among tasks allows a lower number of total parameters in the networks (Caruana, 1997). This reduces the memory footprint and inference time.

Despite the advantages of MTL, the application of this paradigm to real-world scenarios poses several challenges. During training, MTL requires to integrate the supervisory signals of multiple tasks into the same neural network. These supervisory signals, which are the gradients of the different training losses with respect to the network parameters, present varying magnitudes across tasks and parameters. Therefore, following a naive MTL implementation, there is the risk of an uneven contribution of the different tasks to the network training (Misra et al., 2016; Standley et al., 2020). In particular, the tasks with overall larger gradients would contribute more to the parameter updates and would dominate over the tasks with smaller gradients. This uneven contribution of the different tasks to the network training is a key issue in MTL. In practice, this issue is commonly addressed by

* Corresponding author at: Centro de Investigación CITIC, Universidade da Coruña, A Coruña, Spain.
E-mail address: a.suarezh@udc.es (Á.S. Hervella).

https://doi.org/10.1016/j.neunet.2023.11.038
Received 7 March 2023; Received in revised form 16 November 2023; Accepted 16 November 2023
Available online 19 November 2023

using weighting coefficients for the different training losses associated to each task (Vandenhende et al., 2021). In this case, the network training is performed by optimizing a multi-task loss that is defined as the weighted sum of the individual task-specific losses. This approach requires to carefully tune the weighting coefficients in order to achieve successful performance. Nowadays, many works applying MTL rely on a grid search for finding the optimal values of the weighting coefficients. However, with increasingly large network architectures and training datasets, an exhaustive grid search gets increasingly prohibitive. Therefore, there is a growing interest in the development of more efficient alternatives.

Automated methods to dynamically balance the multi-task training have also been proposed (Gong et al., 2019; Vandenhende et al., 2021). Most of these methods dynamically adjust weighting coefficients applied to the training losses. Although, equivalently, some methods also explored to directly weight the gradient of each task in the shared layers of the network (Sener & Koltun, 2018). Regarding the criteria to estimate the optimal weights, the Uncertainty was used in Kendall, Gal, and Cipolla (2018), prioritizing tasks with lower inherent noise in their predictions. Meanwhile, other methods prioritize the tasks with lower performance (Guo, Haque, Huang, Yeung, & Fei-Fei, 2018) or the ones that improve through training at a lower rate, such as DWA (Liu, Johns, & Davison, 2019). In contrast, GradNorm (Chen, Badrinarayanan, Lee, & Rabinovich, 2018) aims to estimate the weights that make the gradients of the different tasks more similar in magnitude. In comparison with the standard grid search approach, these methods present the advantage of dynamically balancing the tasks, hence they can potentially adapt to changing conditions during training. However, their capacity to balance the contribution of the different tasks to the training is limited by the use of a single weighting factor per task. In this regard, it is worth noting that the balance among the gradients of the different task can vary across the different parameters of the network. In this scenario, existing methods based on loss-weighting schemes cannot ensure a well-balanced training for all the parameters simultaneously.

In this work, we propose a novel method to balance the training of multiple tasks. The aim of our proposal is to provide a balanced contribution of the different tasks to the training of each individual parameter in a neural network. In order to achieve this, we take as reference the adaptive optimization algorithms that are commonly used in deep learning (Kingma & Ba, 2015; Tieleman & Hinton, 2012). In that regard, it is worth noting that, in a single-task setting, there are usually uneven gradient magnitudes across network parameters. In consequence, some parameters may be trained at a higher rate than others. However, adaptive algorithms, such as Adam (Kingma & Ba, 2015) or RMSprop (Tieleman & Hinton, 2012), dynamically adapt the effective learning rate for each individual parameter, such that larger effective learning rates are used for parameters with usually smaller gradient magnitudes and vice versa. This provides a well-balanced training across parameters. Our proposal, denoted as Multi-Adaptive Optimization (MAO), extends this idea to the multi-task setting, where there are uneven gradient magnitudes across both parameters and tasks. In that regard, instead of using per-task weighting coefficients that are applied to either the training losses or the gradient, MAO dynamically adapts the effective learning rate for each task at each parameter of the network. This results in a well-balanced training across parameters and tasks regardless of the uneven gradient magnitudes. In this work, we integrate our proposal with the Adam optimization algorithm and demonstrate its advantages in the context of MTL for computer vision. Although the proposed approach could be used to balance to training of multiple tasks in any application domain. In summary, the main contributions of our work are the following:

- A novel paradigm to balance the training of multiple tasks is presented. MAO extends the adaptive optimization commonly applied across parameters to be simultaneously applied across both parameters and tasks. This allows automatically balancing any number of tasks such that all of them contribute the same to the training of the network.
- We integrate the proposed approach with the Adam optimization algorithm and demonstrate that it is a superior alternative for MTL in comparison to the combined use of Adam and other task-balancing approaches based on per-task weighting coefficients.
- The advantages of MAO are demonstrated in the context of MTL for computer vision, considering pixel-level and image-level prediction tasks, different network architectures, and different numbers of tasks. All the experiments are performed on public datasets.
- The performed analysis provide relevant insight on how different task-balancing approaches actually affect the contribution of the different tasks to the training of the networks. This allows us to understand the advantages of our proposal as well as the performance of the different methods.

The remainder of the manuscript is organized as follows. First, Section 2 provides a discussion of previous related works, especially task-balancing approaches for the training of DNNs. In Section 3, we present preliminary concepts that allow us to contextualize and motivate the proposed approach. Then, in Section 4, the proposed approach is presented, including the general formulation as well as the particular integration with the Adam algorithm. Further on, in Section 5, we present the experimental evaluation and discuss the obtained results. Finally, Section 6 summarizes the main conclusions derived from our work.

2. Related work

There exist numerous examples of works successfully applying MTL in computer vision. In fact, complete visual understanding of an image or video usually requires to simultaneously solve several related tasks, such as, e.g., segmentation and classification (Ilyas et al., 2022), object detection and segmentation (He, Gkioxari, Dollár, & Girshick, 2017), pose estimation and depth estimation (Almalioglu et al., 2022), or segmentation and depth estimation (Nakamura, Grassi, & Wolf, 2021). Simultaneously, in the literature there are also several works that focus on addressing key challenges in MTL, aiming at improving the neural network architecture or the optimization process. Vandenhende et al. (2021) provide a comprehensive review of these aspects.

Regarding the optimization in MTL, the difficulties are mainly due to the uneven supervision that is provided by the different tasks, which results in an imbalanced training. In that regard, existing approaches aiming at providing a balanced multi-task training are mainly based on the use of loss weighting schemes (Gong et al., 2019; Vandenhende et al., 2021). In particular, these methods propose automated procedures to dynamically estimate the loss-balancing weights during the network training. The main difference among the existing alternatives is the criteria followed to estimate the weights. In the work of Kendall et al. (2018), the balancing weights are parameterized by the task-dependent uncertainty, which is jointly estimated with the network parameters during the training. In practice, this approach prioritizes the tasks with lower inherent noise. In contrast, Guo et al. (2018) propose to prioritize the more difficult tasks. In this case, the difficulty at any point in the training is intuitively measured by using standard performance evaluation metrics. Alternatively, other approaches explicitly focus on the learning dynamics to balance the multi-task training. Particularly, DWA (Liu et al., 2019) aims at balancing the tasks by increasing the relative weight of the losses that decrease at a slower rate and vice versa. Instead, GradNorm (Chen et al., 2018) aims at directly addressing the imbalanced backpropagated gradients by estimating the loss weights that make the task-specific gradients to have similar magnitude. In general, balanced gradients directly translates into balanced parameter updates, which makes GradNorm the method most closely related to our proposal. However, in contrast to GradNorm, our proposal does not require to estimate any loss weights. Instead, we directly compute per-parameter balanced updates by following a multi-adaptive approximation to the MTL optimization.

Alternatively, some works address the training of multi-task networks as a multi-objective optimization. Particularly, Sener and Koltun (2018) propose to estimate balancing weights that provide a Pareto optimal solution. Meanwhile, Nakamura et al. (2021) propose a greedy method to improve the selection of the most adequate solution in the Pareto front. Their approach aims to obtain a more adequate trade-off between very imbalanced tasks. Additionally, in the case of Sener and Koltun (2018), to avoid affecting the task-specific layers in the network, the loss weights are only applied to the shared layers. Similarly, given that the proposed multi-adaptive approach directly balances the updates on each parameter, our proposal also only affects the shared layers in the network. In that regard, a key difference of our proposal with respect to all the previous methods is precisely that the balance among tasks is independently adjusted at each individual parameter. This ensures a balanced training throughout the whole network.

The imbalanced training in MTL can be aggravated by the conflicting supervision among different tasks. The conflicting supervision is manifested in the form of gradients with opposite sign for the same parameters. This issue has been specifically addressed by some works. In particular, in Yu et al. (2020), gradient conflicts were mitigated by substituting one of the gradients by its projection onto the normal plane of the other. Alternatively, Chen et al. (2020) avoid aggregating gradient components of different sign by masking out those of lower magnitude. These kinds of approaches can be applied together with existing task-balancing methods (Yu et al., 2020), and could potentially be integrated with our proposal too. However, this work is focused on the comparison of the proposed method with other task-balancing methods. Thus, the study of additional conflict mitigation techniques is out of the scope of this work.

3. MTL formulation and contextualization

In the context of deep learning, MTL can be defined as the simultaneous learning of several tasks performed by the same neural network. In particular, given a set of N tasks $\{T_n\}_{n=1}^{N}$, with its associated set of N losses $\{\mathcal{L}_n\}_{n=1}^{N}$, the multi-task network training is performed by the simultaneous optimization of the different task-specific losses.

3.1. Classical MTL optimization

In general, the optimization process in deep learning is performed by a gradient descent algorithm (Goodfellow, Bengio, & Courville, 2016). In particular, during the network training, the network parameters are iteratively updated according to the following rule:

$$\theta_{t+1} = \theta_t + \Delta\theta_t \tag{1}$$

where $\Delta\theta_t$ denotes the update for parameter θ at time t. Following the simplest form of gradient descent, the network parameter updates can be computed as:

$$\Delta\theta_t = -\eta\, g_{\theta_t} \tag{2}$$

where η denotes the learning rate and $g_{\theta_t} = (\partial\mathcal{L}/\partial\theta)_t$ the gradient component for parameter θ at time t due to the training loss $\mathcal{L}$.

In previous works (Vandenhende et al., 2021), the MTL formulation is commonly simplified by defining a new multi-task loss consisting in a weighted sum of the individual task-specific losses. Thus, an integrated multi-task training loss is obtained as:

$$\mathcal{L} = \sum_{n}^{N} \omega_n \mathcal{L}_n \tag{3}$$

where ω_n denotes the weighting coefficient for task T_n. In this case, taking into consideration the rules of differentiation, it is possible to define the network parameter updates in terms of the task-specific gradients. Particularly, the updates can be computed as:

$$\Delta\theta_t = -\eta \sum_{n}^{N} g_{\theta_{n,t}} \tag{4}$$

where $g_{\theta_{n,t}} = (\omega_n \partial\mathcal{L}_n/\partial\theta)_t$ denotes the weighted gradient component for parameter θ at time t due to task T_n.

As it is shown in Eq. (4), the contribution of each task to the parameters updates directly depends on the magnitude of its gradient at each particular parameter. In this regard, the tasks with overall larger gradients will have a greater influence in the direction of training and vice versa. Following the usual MTL formulation with an integrated multi-task loss, the balance among tasks can be adjusted by changing the gradients magnitude through the loss weighting coefficients $\{\omega_n\}_{n=1}^{N}$. However, these weighting coefficients do not allow to balance the task-specific gradients at each particular parameter individually.

3.2. Adaptive optimization algorithms

Nowadays, adaptive optimization algorithms (such as Adam Kingma & Ba, 2015, RMSprop Tieleman & Hinton, 2012, or AdaGrad Duchi, Hazan, & Singer, 2011) are commonly used for the training of DNNs. These algorithms adapt the learning rate for each particular parameter of a neural network, resulting in overall updates of similar magnitude for all the parameters. In this regard, it must be noticed that in a standard gradient descent algorithm the learning rate remains constant throughout the network, hence the parameter updates magnitude is always proportional to the gradients magnitude (see Eq. (2)). In this scenario, the parameters that receive stronger gradients will also be more updated. Conversely, parameters with usually mild gradients will be barely updated. This represents an imbalanced learning across parameters. In contrast, the adaptive algorithms compute an effective per-parameter learning rate that compensates for the varying magnitude of the gradients. This results in a balanced learning across parameters.

Building upon standard gradient descent, the parameters updates using adaptive optimization are computed as:

$$\Delta\theta_t = -\eta_{\theta_t} g_{\theta_t} \tag{5}$$

where η_{θ_t} denotes the effective learning rate for parameter θ at time t. Without loss of generality, these effective per-parameter learning rates are broadly defined as:

$$\eta_{\theta_t} = \frac{\eta}{f(g_{\theta_t}, g_{\theta_{t-1}}, \dots, g_{\theta_0})} \tag{6}$$

where f is a normalization function dependent on current and past gradients for each particular parameter. Taking into consideration Eqs. (5) and (6), it can be seen that the per-parameter learning rates provide an implicit normalization of the gradients at the parameter level. This allows a balanced training across parameters regardless of the backpropagated gradients magnitude.

The definition of function f in Eq. (6) is specific to each particular adaptive optimization algorithm and represents one of the main differences among existing alternatives (Duchi et al., 2011; Tieleman & Hinton, 2012). Additionally, it should be noted that some particular optimization algorithms may also include additional terms dependent on past gradients for the calculus of the parameters updates in Eq. (5), either by substituting the gradient component g_{θ_t}, e.g. Kingma and Ba (2015), or by adding momentum (Goodfellow et al., 2016).

4. Multi-adaptive optimization for MTL

The proposed approach aims to achieve a balanced contribution of the different tasks to the training of each individual parameter in a neural network. In order to do that, MAO extends the adaptive optimization paradigm, previously applied at the parameter-level, to the task-level. Particularly, the adaptive optimization is simultaneously applied across parameters and tasks by computing effective task-specific per-parameter learning rates. These effective learning rates compensate for the varying magnitude of the gradients across parameters and across tasks. Thus, not only all the parameters receive updates of similar

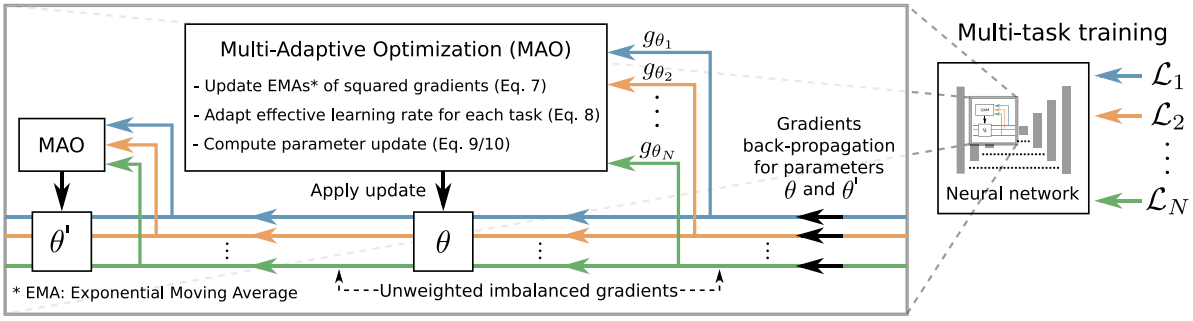


Fig. 1. Proposed approach for the simultaneous training of multiple tasks using Multi-Adaptive Optimization (MAO). The diagram represents the back-propagation and optimization for two different parameters (θ and θ') in a multi-task scenario with N training losses.

magnitude, but also all the tasks present a similar contribution to the parameters updates during the network training. As normalization function to compute the effective learning rates, we use the exponential moving average of squared gradients, which was successfully applied in Adam (Kingma & Ba, 2015) and RMSprop (Tieleman & Hinton, 2012) for balancing the training across parameters. In the case of MAO, the same normalization function is used to balance the training across both parameters and tasks. The proposed method is summarized in the diagram of Fig. 1 whereas the corresponding pseudocode is provided in Algorithm 1.

Algorithm 1 Proposed approach for the simultaneous training of multiple tasks using Multi-Adaptive Optimization (MAO)

Require: Initial parameter vector θ_0, global learning rate η, and N tasks $\{T_n\}_{n=1}^{N}$.
  $t \leftarrow 0$ (Initialize timestep)
  for each task T_n **do**
    $E[g^2_{\theta_n}]_t \leftarrow 0$ (Initialize EMA of squared gradients for task T_n)
  end for
  while stopping criteria is not met **do**
    $t \leftarrow t + 1$ (Forward pass through the network)
    for each task T_n **do**
      $g_{\theta_{n,t}} \leftarrow$ Back-propagate gradients of training loss for task T_n
      $E[g^2_{\theta_n}]_t \leftarrow$ Update EMA of squared gradients for task T_n (Eq. (7) using $g^2_{\theta_{n,t}}$ and $E[g^2_{\theta_n}]_{t-1}$)
      $\eta_{\theta_{n,t}} \leftarrow$ Compute effective learning rates for task T_n (Eq. (8) using η and $E[g^2_{\theta_n}]_t$)
    end for
    $\Delta\theta_t \leftarrow$ Compute parameter updates (Eq. (9) / (10) using $g_{\theta_{n,t}}$ and $\eta_{\theta_{n,t}}$ from tasks $\{T_n\}_{n=1}^{N}$)
    $\theta_t \leftarrow \theta_{t-1} + \Delta\theta_t$
  end while

The process followed by MAO at each training iteration t is as follows. First, for each parameter θ and task n, MAO computes the exponential moving average of the squared gradient component for that parameter $E[g^2_{\theta_n}]_t$. In that regard, the exponential moving average for any given variable x is defined as:

$$E[x]_t = \beta E[x]_{t-1} + (1-\beta)x_t \tag{7}$$

where β denotes the exponential decay rate of the moving average. Particularly, the square root of the moving average $(E[g^2_{\theta_n}]_t)^{1/2}$ is a statistical measure of the gradient magnitude of task n throughout the training process, placing greater importance in the gradient magnitude of more recent iterations. Intuitively, this value represents how strong is the supervisory signal of task n at parameter θ.

The second step is to compute the effective learning rate for each task n and parameter θ. In this regard, the aim is to adapt the learning rate such that all the tasks contribute the same to the training of each parameter, regardless of the uneven strength of the supervisory signals. This is achieved by normalizing the global learning rate with respect to the strength of the supervisory signal, which is given by the term $(E[g^2_{\theta_n}]_t)^{1/2}$. Thus, we obtain task-specific effective learning rates that are computed as:

$$\eta_{\theta_{n,t}} = \frac{\eta}{\sqrt{E[g^2_{\theta_n}]_t + \epsilon}} \tag{8}$$

where η denotes the global learning rate and ϵ is a constant value to avoid the denominator getting too close to zero.

The third step is to compute the parameter updates. Applying gradient descent, the update for a given parameter θ at iteration t is defined as:

$$\Delta\theta_t = -\sum_{n}^{N} \eta_{\theta_{n,t}} g_{\theta_{n,t}} \tag{9}$$

In this regard, the parameter update is obtained by aggregating the individual contributions of the different tasks (i.e. the products $\eta_{\theta_{n,t}} g_{\theta_{n,t}}$). The effective learning rates $\eta_{\theta_{n,t}}$ ensure that, throughout training, all the tasks contribute in a similar amount to the updates of each parameter. However, at any particular iteration, each task can provide a different contribution depending on the magnitude of $g_{\theta_{n,t}}$ at that particular point in time. Additionally, as the gradient component $g_{\theta_{n,t}}$ is a signed value, the magnitude of the final update will depend on the level of a agreement among the different tasks regarding the direction of training (i.e. the sign of the update, either positive or negative). Contributions with the same sign will sum up to a larger update, whereas contributions with opposite sign can counteract each other and will yield a smaller update.

Finally, it is worth noting that our proposal can be integrated with different optimization algorithms. The presented formulation is the result of building our proposal upon standard gradient descent. However, for the experiments in this work, we integrate the proposed multi-adaptive optimization strategy with the Adam (Kingma & Ba, 2015) algorithm, which is one the most commonly used adaptive optimization algorithms in deep learning. Additionally, during the optimization process, we also use weight decay as regularization technique. The adaption of our proposal for using the Adam algorithm and weight decay regularization is explained in detail below.

4.1. MAO using Adam

In order to adapt our proposal to follow the formulation of Adam, it is necessary to change the definition of the parameter updates given in Eq. (9). In particular, for the calculus of the updates, Adam substitutes the gradient component $g_{\theta_{n,t}}$ by a function of the current and past gradients. This function consists in the exponential moving average of the gradients, i.e. $E[g_{\theta_n}]_t$ for parameter θ and task n at time t (see Eq. (7)). Additionally, following the formulation of Adam, two initialization bias correction terms are included. These terms are used to correct the bias in the exponential moving averages due to their

initialization with zeros. Finally, the parameter updates for MAO using Adam are defined as:

$$\Delta\theta_t = -\sum_n^N \eta_{\theta_{n,t}} E[g_{\theta_n}]_t \frac{1-\beta_2^t}{1-\beta_1^t} \tag{10}$$

where β_1 denotes the exponential decay rate for $E[g_{\theta_n}]_t$ and β_2 the exponential decay rate for $E[g^2_{\theta_n}]_t$, which is included in the calculus of $\eta_{\theta_{n,t}}$ (see Eq. (8)).

4.2. MAO using weight decay

Weight decay is a regularization technique that decays the network parameters in a way proportional to their own magnitude. In the literature, this technique has been typically implemented as L2 regularization. However, this approximation would affect the calculus of the task-specific per-parameter learning rates in our proposal. Thus, in order to keep the intended balance across parameters and tasks, we follow (Loshchilov & Hutter, 2019) and apply weight decay by directly modifying the parameter updates defined in Eq. (9). In this case, the parameter updates are obtained as:

$$\Delta\theta_t = -\sum_n^N \eta_{\theta_{n,t}} g_{\theta_{n,t}} \; - \eta\lambda\theta \tag{11}$$

where λ denotes the weight decay coefficient. Similarly, when using both Adam and weight decay, the parameter updates are obtained as:

$$\Delta\theta_t = -\sum_n^N \eta_{\theta_{n,t}} E[g_{\theta_n}]_t \frac{1-\beta_2^t}{1-\beta_1^t} \; - \eta\lambda\theta \tag{12}$$

Finally, regarding the computational complexity of the proposed approach, the time complexity of the algorithm is linear with respect to the number of tasks. This is due to the gradients and parameter updates that are independently computed for each task. This applies both for the general approximation using Eq. (9) as well as for the integration of our proposal with Adam using Eq. (10). With regards to the default values of the different hyperparameters in our proposal, we follow common practices for closely related algorithms, such as Adam or RMSprop. In particular, we use $\epsilon = 1e\text{-}8$ as the small constant for Eq. (8), $\beta_1 = 0.9$ as the exponential decay rate for the moving average of gradients $E[g_{\theta_n}]_t$, and $\beta_2 = 0.999$ as the exponential decay rate for the moving average of squared gradients $E[g^2_{\theta_n}]_t$. Meanwhile, the value of the global learning rate η and the weight decay coefficient λ will depend on the particular application.

5. Experiments and results

The proposed approach is validated and studied in the context of MTL for computer vision. In that regard, we perform an exhaustive experimentation with different multi-task settings and real-world datasets, including both image-level and pixel-level prediction tasks. Additionally, in order to evaluate the performance of our proposal with respect to the state-of-the-art, the same experimentation is also performed for several alternative approaches. Particularly, in our experimentation, we include Uncertainty Weighting (Kendall et al., 2018), DWA (Liu et al., 2019), and GradNorm (Chen et al., 2018), which are the three best performing approaches in the experimental analysis performed in Vandenhende et al. (2021). Additionally, we also include an additional alternative where all the tasks are uniformly weighted (Equal Weights). In order to perform a fair comparison among the different methods, we follow (Vandenhende et al., 2021) and use the best learning rate for each method in each dataset. To take into account the stochasticity of the training process, all the experiments are performed with 5 repetitions using 5 different random seeds. In that regard, all the quantitative results are reported as mean ± standard deviation. Additionally, to assess whether the difference between the best method and the others is statistical significant, we perform a Student's t-test with paired samples for each metric.

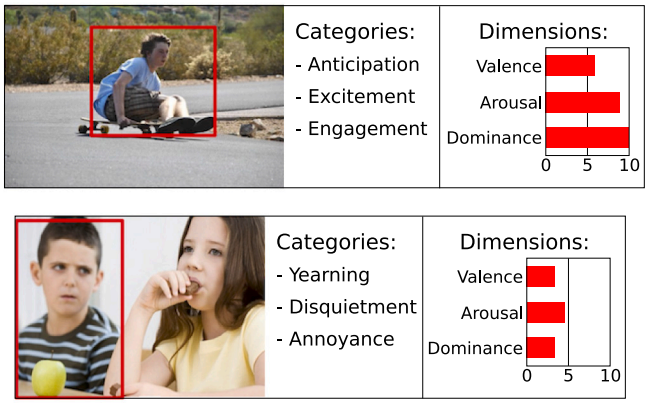


Fig. 2. Examples of input images and ground truth for EMOTIC (Kosti et al., 2020). The bounding box for the person that is being analyzed is draw in red.

5.1. Image-level prediction tasks

5.1.1. Dataset and tasks

For the image-level prediction tasks, we use the emotion recognition dataset EMOTIC (Kosti, Alvarez, Recasens, & Lapedriza, 2020). This public dataset contains images of people in real-world environments annotated with their apparent emotions. The dataset includes annotations for both multi-label classification and regression tasks. Particularly, there are 26 non mutually exclusive emotion categories and 3 continuous emotion dimensions. Additionally, the bounding boxes for the people in the images are also provided. The images, of varying sizes, are resized to a common size of 224 × 224 pixels for convenience. The dataset contains 23,571 images with 34,320 annotated people and provides a standard split of 70% training, 10% validation, and 20% test. Fig. 2 depicts some representative examples of the images in this dataset and the different tasks.

5.1.2. Network architecture and training objectives

We use the network architecture proposed in Kosti, Alvarez, Recasens, and Lapedriza (2017) for addressing this same multi-task scenario. In general, this architecture follows the usual approach in the literature for image-level prediction tasks i.e. using a convolutional encoder followed by one or more fully connected layers. However, this network consists of two encoders, one that receives the whole image as input and other that receives the bounding box of each particular person. The output of both encoders is concatenated and fed to a series of fully connected layers that generate the final predictions. For uniformity among the experiments, we use ResNet-18 (He, Zhang, Ren, & Sun, 2016) as backbone, i.e. as the architecture for the two encoders in the network.

For multi-label classification, the network predicts a vector containing the probabilities of all the discrete categories. A sigmoid function is applied in the last layer of the network to generate the bounded probabilities. Then, the training objective is the minimization of the binary cross-entropy between predicted and ground truth probabilities. For the regression task, the network directly predicts the three independent dimensions using a linear output layer. The training objective is the minimization of the L1-norm of the difference between prediction and ground truth. In summary, the standard setting of EMOTIC has 2 independent losses (using binary cross-entropy and L1-norm). However, we also use the EMOTIC dataset to study the performance of the different task-balancing methods with a higher number of tasks. This alternative setting is constructed by addressing the multi-label classification as a MTL problem. In this case, each category represents an individual task and there are 26 independent losses (using binary cross-entropy).

Table 1
Results with different learning rates on the EMOTIC dataset. Underline denotes the best result of each method. The relative improvement of the best result with respect to the result given by the alternative learning rate is indicated between parentheses. LR denotes Learning Rate.

Method	LR	Categories (mAP ↑)	Dimensions (MAE ↓)
Equal Weights	1e−3	23.69	0.9975 (3.45%)
	1e−4	25.11 (6.00%)	1.0332
Uncertainty	1e−3	24.32	1.0151 (1.46%)
	1e−4	25.48 (4.77%)	1.0301
DWA	1e−3	23.96	0.9981 (2.65%)
	1e−4	25.26 (5.43%)	1.0253
GradNorm	1e−3	24.1	0.9940 (4.22%)
	1e−4	25.41 (5.44%)	1.0378
MAO	1e−3	25.82 (2.91%)	0.9628 (2.16%)
	1e−4	25.09	0.9841

5.1.3. Training details

The training methodology is mainly based on the proposal of Kosti et al. (2017). In particular, both encoders of the network are pre-trained in a supervised image classification task. The encoder that receives the whole images as input is pre-trained on the Places dataset (Zhou, Lapedriza, Xiao, Torralba, & Oliva, 2014), whereas the encoder that receives the bounding box of each person is pre-trained on the ImageNet dataset (Deng et al., 2009). The optimization of the baseline methods is conducted with the Adam algorithm (Kingma & Ba, 2015), adopting the standard decay rates of $\beta_1 = 0.9$ and $\beta_2 = 0.999$ as well as the standard small constant $\epsilon =$ 1e-8. Also, weight decay with coefficient $\lambda = 0.0005$ is applied (Loshchilov & Hutter, 2019). The same hyperparameters are used for MAO. The training is conducted with a batch size of 52 images for a duration of 15 epochs. A reduction of the learning rate by a factor of 10 is applied after the first half of training (Kosti et al., 2017). Regarding the initial learning rate, we followed (Vandenhende et al., 2021) and selected the best initial learning rate α for each method. We performed an initial grid search in the set {1e-5, 1e-4, 1e-3} and progressively extended the range if the best value was any of the extremes. We found that $\alpha =$ 1e-3 provided the best results for our proposal in both tasks, Categories and Dimensions. For the other methods, $\alpha =$ 1e-4 provided the best results for Categories, whereas $\alpha =$ 1e-3 provided the best results for Dimensions. In these cases, we used $\alpha =$ 1e-4, as it provided the best overall performance. The differences between using $\alpha =$ 1e-3 or $\alpha =$ 1e-4 according to our preliminary experiments can be seen in Table 1. As data augmentation, we use random horizontal flipping of the images.

5.1.4. Evaluation metrics

The quantitative evaluation is performed using common metrics in the literature (Kosti et al., 2020; Najar & Bouguila, 2022). In particular, multi-label classification is evaluated by means of mean Average Precision (mAP) and mean F1-score (mF1). The regression is evaluated using Mean Average Error (MAE) and Mean Squared Error (MSE).

5.1.5. Results for EMOTIC

Table 2 depicts the results of the experiments on the EMOTIC dataset. First of all, the obtained results show that MAO outperforms all the other alternatives in these experiments, obtaining significantly better results in both tasks. The improvement is especially notable for Dimensions. However, it must be noticed that, in this case, the differences between tasks are also affected by the selection of the learning rate for the baseline methods. As seen in Table 1, other methods present a compromise between the performance in Categories and the performance in Dimensions. In this case, it happens that the value that provides the best overall performance is less beneficial for the task Dimensions, which explains the greater improvement of MAO in this task. It is worth noting that MAO does not present such compromise, hence it is able to simultaneously achieve the best possible result for each task. This makes it possible for MAO to always outperform the other alternatives, regardless of the hyperparameter selected for those methods.

5.1.6. Multi-label classification as multi-task learning

In order to study the performance of the different methods when facing a high number of tasks, we reformulate the multi-label classification in EMOTIC as a MTL problem. For these experiments, we only use the categorical annotations in this dataset and consider each of the 26 categories as an individual task. Then, the different task-balancing methods are used to balance the learning among categories/tasks.

Table 3 depicts the results of the experiments with 26 tasks. The results show that, in this challenging setting, MAO significantly outperforms all the other alternatives. This indicates that the proposed method is robust and able to perform well regardless of the number of tasks. Additionally, MAO is the only method for which this experiment reaches the same performance than the standard 2-task setting of EMOTIC. In that regard, it should be noticed that the result achieved by Equal Weights in this experiment ($24.29 \pm 0.17\%$ mAP) corresponds to the single-task baseline for Categories. Thus, it can be seen in Table 2 that all the methods are able to improve the performance for Categories by including Dimensions as an additional complementary task. However, MAO is able to achieve the same results by only exploiting the relations among the 26 categories, without requiring any additional annotations (such as those corresponding to the regression task, i.e. Dimensions).

5.2. Pixel-level prediction tasks

5.2.1. Dataset and tasks

For the pixel-level prediction tasks, we use the public dataset NYUDv2 (Silberman, Hoiem, Kohli, & Fergus, 2012). This dataset is focused on indoor scene understanding and provides a multi-task scenario consisting of semantic segmentation, monocular depth estimation, and surface normal prediction. In particular, the dataset provides images of common scenes in different room types, such as living rooms, bedrooms, kitchens, etc. The depth maps are obtained from a Microsoft Kinnect device and the surface normals are automatically derived from the depth data (Eigen & Fergus, 2015). The semantic segmentation task comprises 40 different classes. The size of the images is 480×640 pixels and the dataset provides 795 training images and 654 validation images. Fig. 3 depicts some representative examples of the images in the dataset as well as the different tasks.

5.2.2. Network architecture and training objectives

In this case, given the variety of alternatives in the literature, we use two different state-of-the-art network architectures in our experiments: Multi-Task Attention Network (MTAN) (Liu et al., 2019) and Multi-Scale Task Interaction Network (MTI-Net) (Vandenhende, Georgoulis, & Van Gool, 2020). These two networks are exemplar of the two main trends regarding the architectural approaches to MTL (Vandenhende et al., 2021). Thus, we ensure that our experiments are relevant to the broad spectrum of current practices.

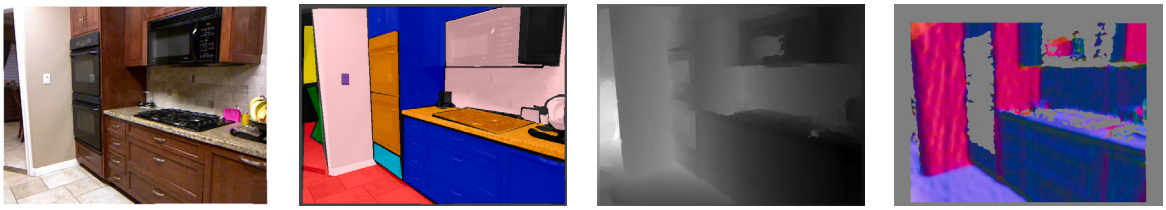
With regards to the characteristics of the two networks, MTAN consists of a shared backbone and several task-specific heads with attention modules. The attention modules aim at selecting the most relevant features for each task from the global feature pool learned in the shared backbone. Similarly, MTI-Net also presents a shared backbone followed by several task-specific heads. However, in this case, the heads make several initial task predictions at multiple scales. Then, the learned task-specific features are distilled at every scale and finally aggregated across scales to make the final predictions. In both networks, the shared backbone is an standard convolutional encoder. For uniformity among the experiments, we use ResNet-18 (He et al.,

Table 2
Results for EMOTIC (multi-label classification with 26 emotion and regression with 3 dimensions). Bold denotes the best result for each evaluation metric in terms of mean value. Asterisks denote whether the difference between the best result and the others is statistically significant (∗ denotes p value < 0.05; ∗∗ denotes p value < 0.01).

Method	Categories ↑		Dimensions ↓	
	mAP	mF1	MAE	MSE
Equal weights	25.35 ± 0.15	29.88 ± 0.56	1.0272 ± 0.0073	1.7139 ± 0.0182
Uncertainty (Kendall et al., 2018)	25.31 ± 0.13	29.74 ± 0.40	1.0254 ± 0.0055	1.7041 ± 0.0176
DWA (Liu et al., 2019)	25.23 ± 0.11	29.51 ± 0.61	1.0280 ± 0.0064	1.7154 ± 0.0117
GradNorm (Chen et al., 2018)	25.32 ± 0.16	29.62 ± 0.21	1.0273 ± 0.0100	1.7070 ± 0.0261
MAO (Proposed)	**25.92 ± 0.09	*30.31 ± 0.34	**0.9557 ± 0.0050	**1.4942 ± 0.0163

Table 3
Results for EMOTIC-Categories (multi-label classification with 26 categories). Evaluation by means of AP(%) (higher is better). Bold denotes the best result for each category/task in terms of mean value. Asterisks denote whether the difference between the best result and the others is statistically significant (∗ denotes p value < 0.05; ∗∗ denotes p value < 0.01).

Category/Task	Equal	Uncert	DWA	GNorm	MAO
Affection	28.89 ± 0.87	28.80 ± 0.69	29.20 ± 0.68	29.27 ± 0.75	*31.20 ± 0.42
Anger	9.26 ± 0.93	9.39 ± 0.84	8.96 ± 0.83	11.47 ± 0.93	10.62 ± 0.28
Annoyance	12.99 ± 0.84	13.11 ± 0.56	12.81 ± 0.84	13.32 ± 0.92	13.79 ± 0.23
Anticipation	54.08 ± 0.82	54.48 ± 0.33	54.24 ± 0.75	55.64 ± 0.26	**56.57 ± 0.12
Aversion	5.28 ± 0.56	5.25 ± 0.19	5.37 ± 0.58	5.51 ± 0.22	*7.10 ± 0.44
Confidence	65.69 ± 0.55	65.95 ± 0.51	65.92 ± 0.51	70.13 ± 0.80	**75.27 ± 0.13
Disapproval	10.85 ± 0.80	11.34 ± 0.56	10.82 ± 0.79	12.12 ± 1.08	12.47 ± 0.30
Disconnection	23.21 ± 1.08	23.46 ± 0.62	23.30 ± 1.05	23.68 ± 0.56	24.47 ± 0.39
Disquietment	17.48 ± 0.51	17.16 ± 0.59	17.52 ± 0.56	17.23 ± 0.22	16.79 ± 0.26
Doubt/Confusion	17.67 ± 0.36	17.78 ± 0.48	17.66 ± 0.35	17.53 ± 0.34	17.25 ± 0.11
Embarrassment	2.17 ± 0.20	2.18 ± 0.20	2.07 ± 0.20	2.19 ± 0.16	*2.65 ± 0.20
Engagement	83.47 ± 0.47	83.41 ± 0.34	83.33 ± 0.38	84.41 ± 0.32	**86.20 ± 0.10
Esteem	15.05 ± 0.53	14.49 ± 0.63	15.07 ± 0.50	14.76 ± 0.31	**16.37 ± 0.16
Excitement	63.80 ± 0.50	63.47 ± 0.32	63.78 ± 0.51	65.26 ± 0.16	**68.76 ± 0.12
Fatigue	10.64 ± 0.42	10.73 ± 0.79	10.45 ± 0.44	11.45 ± 0.57	10.65 ± 0.20
Fear	4.83 ± 0.19	5.02 ± 0.61	4.84 ± 0.19	5.42 ± 0.55	5.78 ± 0.32
Happiness	67.28 ± 0.62	67.25 ± 0.62	67.46 ± 0.38	67.48 ± 0.33	66.69 ± 0.18
Pain	6.98 ± 0.80	7.17 ± 0.44	6.77 ± 0.87	8.29 ± 0.87	7.81 ± 0.78
Peace	21.98 ± 0.84	22.39 ± 0.65	21.97 ± 0.84	22.71 ± 0.57	23.44 ± 0.44
Pleasure	40.59 ± 0.90	41.04 ± 0.73	41.19 ± 0.73	41.69 ± 0.27	42.06 ± 0.16
Sadness	17.69 ± 1.56	17.61 ± 1.69	17.77 ± 1.65	19.95 ± 0.90	20.63 ± 0.90
Sensitivity	5.65 ± 0.39	5.96 ± 0.65	5.61 ± 0.36	6.11 ± 0.44	6.22 ± 0.21
Suffering	17.79 ± 2.02	17.32 ± 1.60	17.59 ± 1.87	19.18 ± 0.95	*21.68 ± 0.65
Surprise	7.16 ± 0.08	7.38 ± 0.61	7.22 ± 0.19	7.78 ± 0.47	*8.97 ± 0.38
Sympathy	12.37 ± 0.51	11.96 ± 0.34	12.38 ± 0.51	12.59 ± 0.58	13.19 ± 0.31
Yearning	8.82 ± 0.40	8.36 ± 0.42	8.75 ± 0.40	8.48 ± 0.26	8.22 ± 0.19
Mean	24.29 ± 0.17	24.33 ± 0.22	24.31 ± 0.18	25.14 ± 0.06	**25.96 ± 0.07



(a) Input image (b) Semantic segmentation (c) Depth estimation (d) Normal prediction

Fig. 3. Example of input image and ground truth of the different tasks in NYUDv2 (Silberman et al., 2012).

2016) as backbone in MTAN and HRNet-18 (Sun, Xiao, Liu, & Wang, 2019) as backbone in MTI-Net.

The training of the different tasks is formulated following the usual approaches in state-of-the-art related works (Liu et al., 2019; Xu, Ouyang, Wang, & Sebe, 2018). For semantic segmentation, the probability of each class is predicted in a different output channel of the network. Then, the normalized probabilities are obtained with a softmax operation and the training objective is the minimization of the cross-entropy between predicted and ground truth probabilities. The monocular depth estimation is approached as a direct regression of the depth values in a single output channel of the network. For this task, the training objective is the minimization of the L1-norm of the difference between predicted and ground truth values. For surface normal prediction, the normal vectors are predicted using a different output channel for each Cartesian component. Then, the training objective is the maximization of the cosine similarity between the predicted and ground truth vectors. In total, there are 3 independent losses for NYUDv2 (using cross-entropy, L1-norm, and cosine similarity).

5.2.3. Training details

Regarding the training methodology, we mainly adopt well-proven practices from previous works (Liu et al., 2019; Xu et al., 2018). In this case, also considering the variety of approaches in the literature, we explore both training from scratch and the use of network backbones that were pre-trained on the ImageNet dataset (Deng et al., 2009). For the experiments training from scratch, the networks are randomly

Table 4
Results for NYUDv2 with MTAN and MTI-Net pretrained on ImageNet. Bold denotes the best result for each evaluation metric in each network architecture in terms of mean value. Asterisks denote whether the difference between the best result and the others is statistically significant (∗ denotes p value < 0.05; ∗∗ denotes p value < 0.01).

Method	Semantic ↑		Depth ↓		Normals ↓	
	mIoU	pAcc	Abs. δ	Rel. δ	Mean	Median
MTAN						
Equal Weights	39.77 ± 0.24	69.38 ± 0.08	0.4017 ± 0.0032	0.1585 ± 0.0016	22.31 ± 0.08	15.27 ± 0.07
Uncertainty (Kendall et al., 2018)	39.85 ± 0.30	69.51 ± 0.12	0.4017 ± 0.0025	0.1579 ± 0.0011	22.16 ± 0.06	15.10 ± 0.05
DWA (Liu et al., 2019)	39.83 ± 0.23	69.34 ± 0.10	0.4029 ± 0.0025	0.1586 ± 0.0012	22.36 ± 0.04	15.33 ± 0.07
GradNorm (Chen et al., 2018)	39.62 ± 0.40	69.20 ± 0.16	0.4020 ± 0.0036	0.1588 ± 0.0008	22.05 ± 0.08	14.98 ± 0.05
MAO (Proposed)	**40.10 ± 0.31**	**69.62 ± 0.15**	**∗∗0.3960 ± 0.0032**	**∗∗0.1537 ± 0.0014**	**∗∗21.28 ± 0.07**	**∗∗14.05 ± 0.06**
MTI-Net						
Equal Weights	**39.07 ± 0.45**	**68.94 ± 0.19**	0.3829 ± 0.0021	0.1501 ± 0.0014	22.62 ± 0.05	16.25 ± 0.09
Uncertainty (Kendall et al., 2018)	38.67 ± 0.32	68.63 ± 0.25	**0.3809 ± 0.0017**	0.1489 ± 0.0020	22.39 ± 0.04	15.96 ± 0.06
DWA (Liu et al., 2019)	38.86 ± 0.36	68.78 ± 0.20	0.3812 ± 0.0014	0.1496 ± 0.0013	22.60 ± 0.03	16.15 ± 0.06
GradNorm (Chen et al., 2018)	38.59 ± 0.29	68.43 ± 0.16	0.3783 ± 0.0032	**0.1475 ± 0.0023**	21.78 ± 0.05	15.26 ± 0.08
MAO (Proposed)	37.94 ± 0.43	68.19 ± 0.10	0.3820 ± 0.0038	0.1484 ± 0.0018	**∗∗21.21 ± 0.05**	**∗∗14.84 ± 0.08**

Table 5
Results for NYUDv2 with MTAN and MTI-Net trained from scratch. Bold denotes the best result for each evaluation metric in each network architecture in terms of mean value. Asterisks denote whether the difference between the best result and the others is statistically significant (∗ denotes p value < 0.05; ∗∗ denotes p value < 0.01).

Method	Semantic ↑		Depth ↓		Normals ↓	
	mIoU	pAcc	Abs. δ	Rel. δ	Mean	Median
MTAN						
Equal Weights	26.21 ± 0.23	58.89 ± 0.16	0.4832 ± 0.0023	0.1874 ± 0.0005	25.82 ± 0.12	18.11 ± 0.15
Uncertainty (Kendall et al., 2018)	25.89 ± 0.53	58.68 ± 0.30	0.4779 ± 0.0014	0.1850 ± 0.0005	25.44 ± 0.09	17.68 ± 0.09
DWA (Liu et al., 2019)	26.22 ± 0.43	58.80 ± 0.15	0.4809 ± 0.0043	0.1865 ± 0.0013	25.78 ± 0.13	18.10 ± 0.13
GradNorm (Chen et al., 2018)	26.20 ± 0.24	58.76 ± 0.13	0.4789 ± 0.0018	0.1853 ± 0.0009	24.92 ± 0.09	17.11 ± 0.09
MAO (Proposed)	**∗∗28.73 ± 0.29**	**∗∗61.29 ± 0.18**	**∗∗0.4496 ± 0.0021**	**∗∗0.1735 ± 0.0008**	**∗∗23.24 ± 0.08**	**∗∗15.32 ± 0.09**
MTI-Net						
Equal Weights	28.16 ± 0.45	60.35 ± 0.27	0.4608 ± 0.0022	0.1783 ± 0.0007	25.85 ± 0.06	19.16 ± 0.06
Uncertainty (Kendall et al., 2018)	27.71 ± 0.32	60.07 ± 0.32	0.4571 ± 0.0034	0.1777 ± 0.0013	25.48 ± 0.13	18.64 ± 0.16
DWA (Liu et al., 2019)	28.34 ± 0.59	60.53 ± 0.37	0.4609 ± 0.0035	0.1793 ± 0.0016	25.87 ± 0.03	19.13 ± 0.07
GradNorm (Chen et al., 2018)	27.92 ± 0.24	59.98 ± 0.37	0.4494 ± 0.0064	0.1736 ± 0.0013	24.52 ± 0.12	17.40 ± 0.15
MAO (Proposed)	**∗∗30.85 ± 0.28**	**∗∗62.50 ± 0.40**	**∗∗0.4324 ± 0.0063**	**∗∗0.1670 ± 0.0027**	**∗∗22.97 ± 0.08**	**∗∗15.84 ± 0.11**

initialized following the method of He, Zhang, Ren, and Sun (2015). The optimization of the baseline methods is conducted with the Adam algorithm (Kingma & Ba, 2015), adopting the standard decay rates of $\beta_1 = 0.9$ and $\beta_2 = 0.999$ as well as the standard small constant $\epsilon =$ 1e-8. Also, weight decay with coefficient $\lambda = 0.0001$ is applied (Loshchilov & Hutter, 2019). The same hyperparameters are used for MAO. The training is conducted with a batch size of 4 images. The duration of training is 100 epochs when using network backbones pre-trained on ImageNet and 170 epochs when training from scratch. This difference is intended to ensure the convergence of the networks in all the cases. Regarding the initial learning rate, we followed (Vandenhende et al., 2021) and selected the best initial learning rate α for each method. We performed an initial grid search in the set {1e-5, 1e-4, 1e-3} and progressively extended the range if the best value was any of the extremes. In this case, a learning rate $\alpha =$ 1e-4 was deemed the most adequate for all the methods, hence this is the value used in all the experiments. During training, a polynomial learning rate decay with exponent $p = 0.9$ is applied (Vandenhende et al., 2021). In order to avoid overfitting, we perform spatial data augmentation in all the experiments. In particular, we adopt the random transformations performed in Xu et al. (2018).

5.2.4. Evaluation metrics

The quantitative evaluation is also performed using common metrics in the literature (Liu et al., 2019; Vandenhende et al., 2021). In particular, semantic segmentation is evaluated by means of mean Intersection over Union (mIoU) and total pixel Accuracy (pAcc). Depth estimation is evaluated using the average Absolute error (Abs. δ) and the average Relative error (Rel. δ). Meanwhile, surface normal prediction is evaluated by means of the Mean and Median angular deviations of the predicted vectors.

5.2.5. Results for NYUDv2

Tables 4 and 5 show the results of the experiments on the NYUDv2 dataset using pretrained networks and training from scratch, respectively. All the experiments are performed for two different neural networks, MTAN and MTI-Net. In general, the obtained results show that MAO produces the best overall performance. In particular, when training from scratch, MAO clearly outperforms all the other alternatives in all the tasks and regardless of the network architecture. Meanwhile, when using pre-trained networks, the overall performance of the different methods is more similar. When using a pretrained MTAN, MAO achieves significantly better results in two of the three tasks, whereas when using a pretrained MTI-Net, MAO achieves significantly better results only in the Normals prediction task. In this case, the improvement in Normals seems to be at the expense of a relatively lower performance in Semantic segmentation. In this regard, it is worth noting that Semantic is also the task for which MAO offers less improvement when using a pretrained MTAN. These results could be explained by a combination of the pretraining especially favoring the Semantic segmentation task and an initial imbalanced scenario in which this task provides stronger supervisory signals in comparison to the other two tasks. This latter question is further studied in Sections 5.2.6 and 5.2.7. Regarding the effect of the pretraining on each task, taking as reference the approach Equal Weights, it can be seen that the pretraining benefits the Semantic segmentation the most (average relative improvement of 45,24% mIoU in comparison to 16.88% for Depth and 13.04% for Normals). This could reduce the beneficial impact of the learned shared features in the Semantic segmentation, given that the pretrained features seem to already be very well aligned with the requirements of this task.

Regarding the network architectures, it is observed in Tables 4 and 5 that MTAN and MTI-Net provide, overall, similar performance.

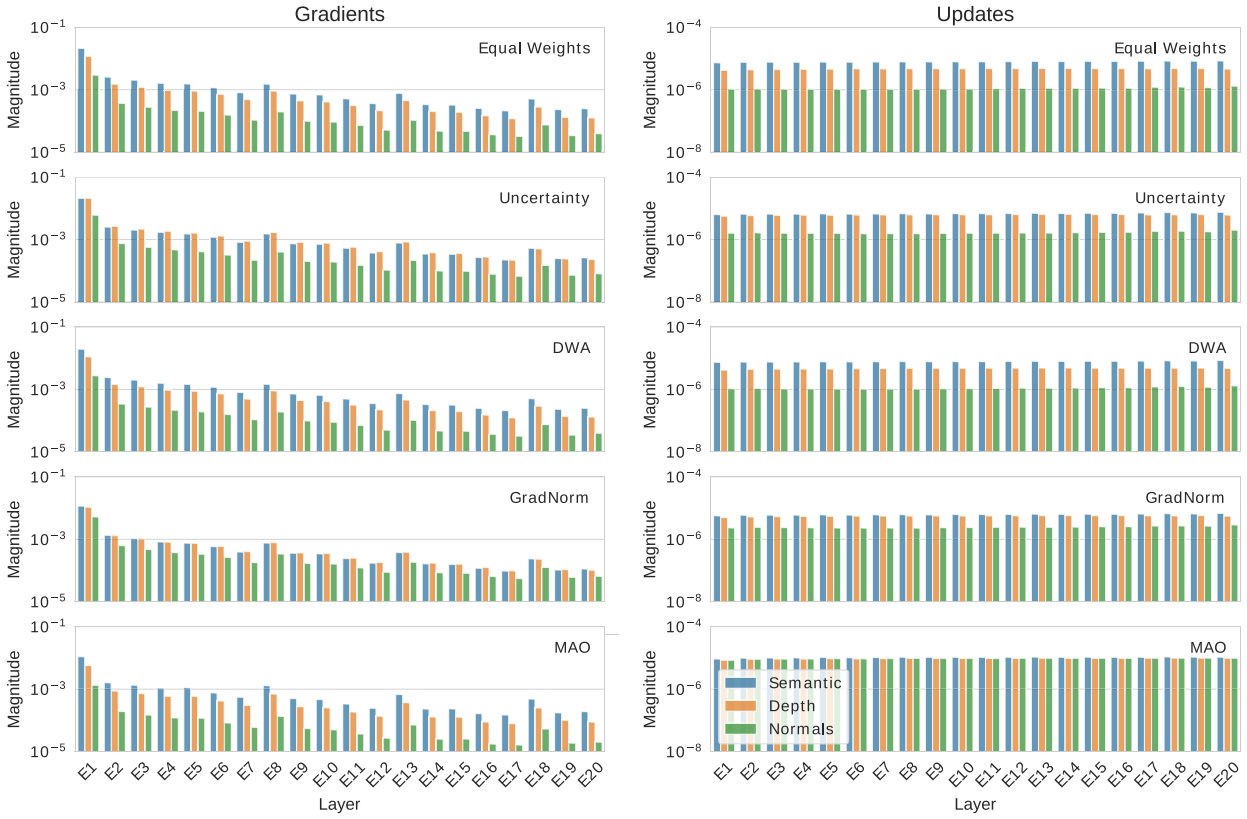


Fig. 4. Layer-wise average magnitude of the task-specific gradients (left) and layer-wise average contribution of each task to the parameter updates (right) during the whole training. Experiments performed using MTAN on the NYUDv2 dataset.

Although there are some particular trends, such as MTI-Net always providing the best results for depth estimation. In contrast, for semantic segmentation, MTI-Net provides the best results when training from scratch but MTAN does so when using pre-trained networks. In general, it seems that different networks may be better suited to different scenarios. These results show that the network architecture also plays a role in the balance among tasks, at least in terms of the final performance. In this regard, it is known that basic decisions such as the number of shared or task-specific layers can affect the obtained results (Bragman, Tanno, Ourselin, Alexander, & Cardoso, 2019). Additionally, in this case, MTAN and MTI-Net also present particular mechanisms that are expected to positively affect the performance in MTL.

5.2.6. Task balance analysis across network layers

To better understand the success of the proposed approach and study the real effect of the different alternatives on the learning process, we perform an in-depth analysis of the learning dynamics within the neural network. This analysis is carried out by measuring the backpropagated gradients during the training and computing a series of relevant metrics. In particular, we calculate the average gradient magnitude of each task at each layer of the network. These average gradient magnitudes represent the overall strength of the supervisory signal provided by each task. We also compute the average contribution of each task to the parameter updates at each layer of the network. These average contributions to the parameter updates represent how much influence each task has on the training. The contribution of each task to the parameter updates is computed as the product of the gradients magnitude and the effective learning rate, following the formulation of Adam. In the case of MAO, we use the computed task-specific learning rates, whereas in the classical loss weighting approaches the same effective learning rate is used for all the tasks.

The described analysis is performed on the NYUDv2 dataset using MTAN. The obtained results are depicted in the graphs of Fig. 4, where each bar represents the averages throughout the whole training. The first observation is that, for all the alternatives, the gradients vary significantly throughout the network whereas the updates are very similar across the different layers. This is a consequence of the adaptive optimization algorithm Adam that is used in all the cases, providing a balanced learning across parameters regardless of the backpropagated gradients magnitudes.

Regarding the relation among tasks, it is observed that the backpropagated gradients are imbalanced in all the cases. For previous methods, this gradient imbalance across tasks is directly translated into imbalanced contributions to the parameter updates. However, the proposed MAO approach is able to provide precisely balanced contributions to the parameter updates in spite of the gradients imbalance. This results in a balanced learning across tasks. This is an important observation that explains the strong results obtained by the proposed approach.

In general, considering all the studied alternatives, the contributions to the parameters updates seem to correlate with the quantitative results depicted in Table 5 for MTAN. For instance, in the reference scenario Equal Weights, depth estimation and normal prediction present lower contributions than semantic segmentation. Then, both Uncertainty and GradNorm visibly increase the contributions of these tasks (see Fig. 4) while also producing an improvement of their quantitative results in Table 5. Moreover, GradNorm is the method providing the highest contributions for normals after MAO and, again, it is also achieving the second best quantitative results for this task in Table 5. These observations indicate that the balance among tasks, in terms of their contribution to the parameter updates, is directly related to the final performance of each task.

5.2.7. Task balance analysis across training steps

Given the non-stationary nature of the network training, we perform an additional analysis considering the evolution of the learning process

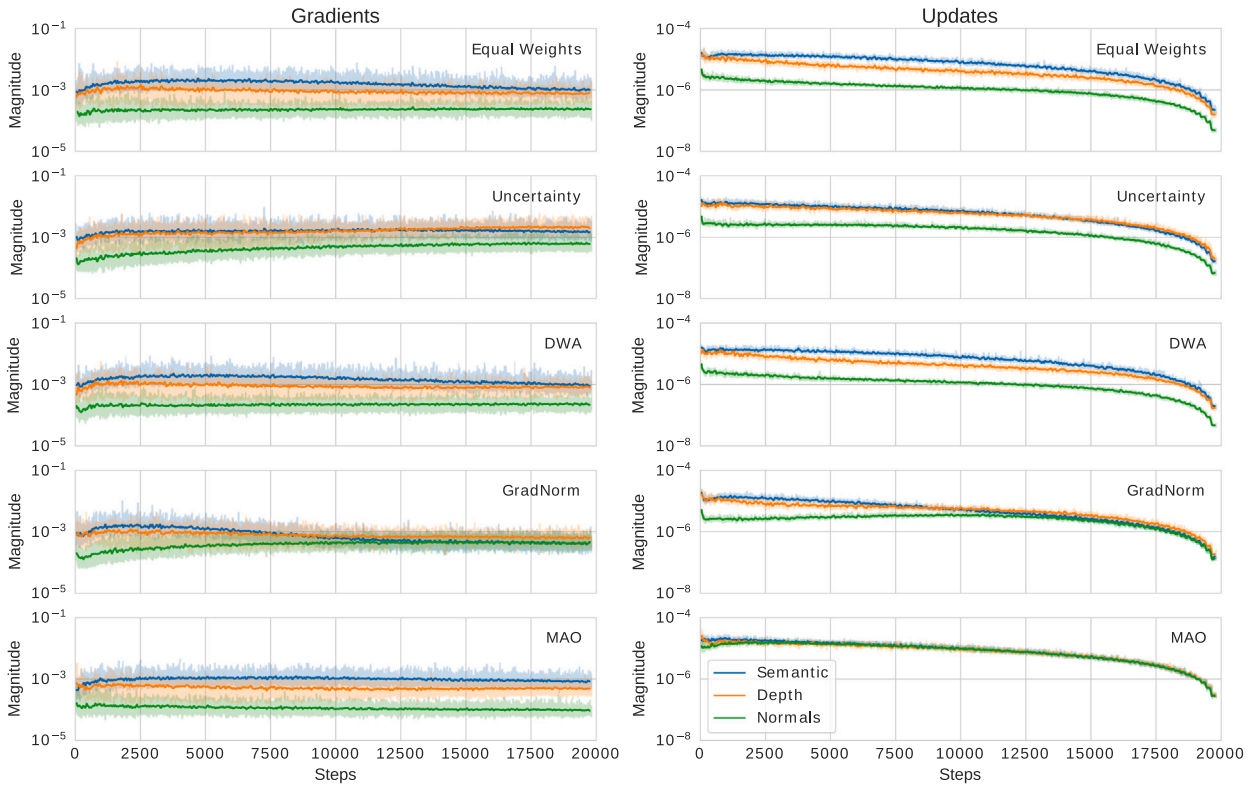


Fig. 5. Average magnitude of the task-specific gradients (left) and average contribution of each task to the parameter updates (right) for every training step. Experiments performed using MTAN on the NYUDv2 dataset.

through time. This analysis is based on the same experiments and metrics that were used in Section 5.2.6. However, in this case, the gradients and contributions to the parameter updates are averaged over the whole network and computed for each training step. The results are depicted in the graphs of Fig. 5. In this case, it must be noticed that the drop-off in computed contributions towards the end of the training is due to the applied learning schedule described in Section 5.2.3.

Regarding the comparison among alternatives, it is observed that, in terms of contributions to the parameter updates, the best balance at any training step is always provided by MAO. In this regard, the proposed approach provides a very tight balance among tasks throughout the whole training. This allows a successful MTL performance as it is demonstrated in the results of Table 5. Interestingly, GradNorm also produces a good balance among tasks –although not as tight as MAO– towards the end of the training. However, the imbalanced scenario during the first half of the training negatively affects the final performance of this approach. In this case, the existing lag is due to the optimization process that is required to estimate the balancing loss weights of GradNorm. In contrast, Uncertainty Weighting is capable of producing an apparently adequate balance throughout the whole training, but only for two of the three tasks involved.

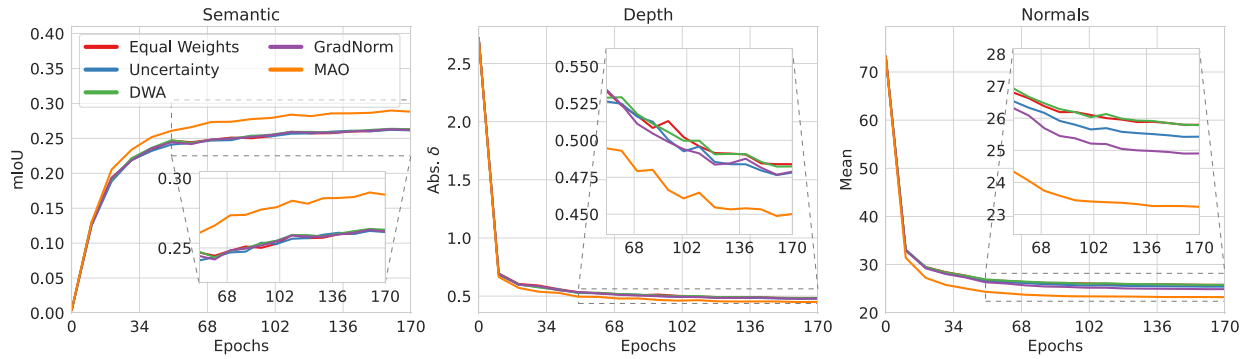
Finally, these results are in line with those reported in Section 5.2.6 and show that a biased task balance during a limited number of epochs can severely influence the aggregated contribution of each task to the training. This demonstrates the importance of achieving an adequate balance among all the tasks and during the whole training.

5.2.8. Evolution of evaluation metrics through training

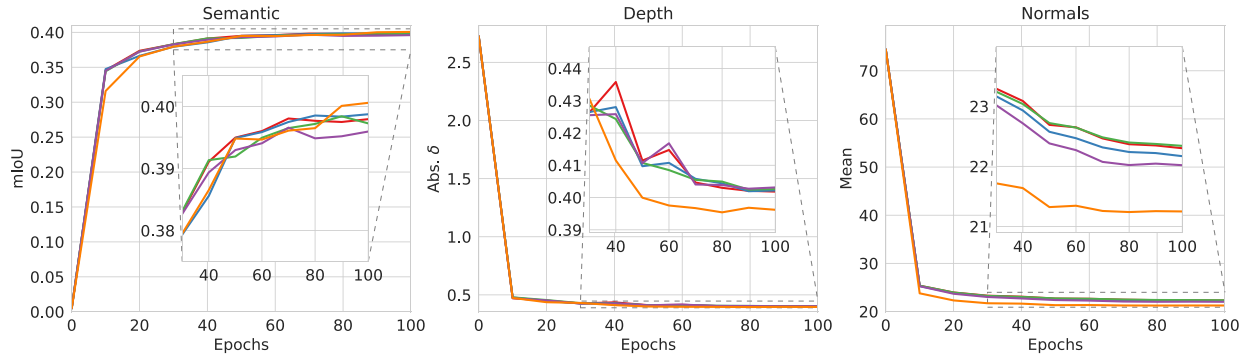
Finally, we also analyze the evolution through the training in terms of evaluation metrics for each of the tasks. These experiments are conducted using the same network as previous analyses (Sections 5.2.6 and 5.2.7). However, given the differences in performance between training from scratch and using pretrained networks, in this case we include both alternatives in the analysis. The results are depicted in the graphs of Fig. 6. It can be seen that the different approaches start to have an effect on the performance of the network since the first epochs of training. Additionally, this effect usually matches the impact of each approach on the final performance of the network. For instance, when training from scratch, our proposal seems to be advantageous for the three tasks since the beginning. This could be explained by the higher quality of the learned features when all the tasks contribute equally to the training. When using a pretrained network, the same advantage is observed for Depth and Normals. However, our proposal slows down the training of Semantic in comparison to the other approaches. This can be explained by two main reasons. First, as seen in Section 5.2.5, the features already provided by the pretraining are especially advantageous for Semantic segmentation, thus reducing the benefit of the shared features learned due to the complementary multitask feedback. Second, as seen in Figs. 4 and 5, our proposal provides a very well balanced training of the different tasks. This necessarily implies a reduction of the relative importance of the tasks that were initially dominating the training (in this case, Semantic segmentation). When training from scratch, the benefits of the high quality shared features outweigh this reduction, but that is not the case when pretrained features very well aligned with the Semantic segmentation task are already available since the beginning of training. In any case, as the training progresses, the multitask shared features seem to be more advantageous and our proposal is able to reach or even surpass other alternatives.

5.3. Impact analysis

In this work, we have performed an extensive experimentation to evaluate and compare the performance of different task-balancing approaches. These experiments, focused on computer vision, have included both image-level and pixel-level prediction tasks, experimental settings with different number of tasks, and different network architectures. Considering all the experiments that were performed, MAO



(a) From scratch



(b) Pretrained

Fig. 6. Evolution of the performance of each task through training. Experiments performed using MTAN on the NYUDv2 dataset.

proves to be the approach that provides the best overall performance. In particular, MAO clearly outperforms the other alternatives in all the experiments except for the one using a pre-trained MTI-Net on NYUDv2. Even in this case, MAO still provides a competitive performance that is on par with the other methods. In this particular case, the differences among methods are due to the tendency of some of them to favor some tasks over others. As it is shown in the analyses of Figs. 4 and 5, there is an imbalanced training scenario in the NYUDv2 dataset, with some tasks providing greater gradients than others. In this scenario, MAO is the only approach able to completely balance the contributions of the different tasks to the training of the network. Comparatively, other approaches have a smaller effect on the balance among tasks. This could explain why the performance of the other task-balancing approaches is usually closer to the performance of Equal Weights and our proposal is the method always producing the most drastic changes in performance. This is likely due to the greater effect of MAO in the balance among tasks. According to Fig. 5, GradNorm is also able to achieve a relatively well balanced contributions to the parameter updates towards the end of the training. This seems to be reflected in some results, such as e.g. the second-best performance in the experiments with 26 tasks on EMOTIC as well as the second-best performance in Normals prediction on NYUDv2. However, in other cases, the results of GradNorm are closer to those of Equal Weights. This seems to indicate that not only it is important to achieve a good balance among tasks, but also it is necessary to do so throughout the whole training. We have demonstrated that this is achieved by the proposed approach.

Finally, regarding potential limitations and challenges for the application of our proposal, it is worth noting that MAO requires to compute the individual backpropagated gradients of each task. Therefore, the complexity of the method is linear with respect to the number of tasks. This can be a limitation when training a high number of tasks. Future works could explore alternatives to improve the computational efficiency in these scenarios. Additionally, regarding future research directions, our analyses provide some relevant results than can be valuable for future works, such as the importance of balancing the contributions of the different tasks throughout the whole training or the interrelation between multi-task learning and the pre-training.

6. Conclusions

In this work, we have presented a novel Multi-Adaptive Optimization (MAO) strategy for automatically producing a task-balanced training in multi-task learning. In particular, our proposal avoids the use of loss weighting schemes and, instead, extends the adaptive optimization paradigm, previously applied at the parameter-level, to the task-level. This allows to achieve a balanced contribution of the different tasks to the network parameter updates, during the whole training and for any number of tasks.

The proposed approach is validated in the context of multi-task learning for computer vision, considering both image-level and pixel-level prediction tasks. In this regard, the experimentation is conducted on two public real-world datasets providing a variety of different scenarios. Additionally, we further studied the performance of the method under different settings, considering relevant factors such as the network architecture and the pre-training of the networks as well as multi-task scenarios with a high number of tasks. All the experiments were performed for several state-of-the-art alternatives, hence providing a valuable comparison of task-balancing approaches. In this regard, the obtained results show that the proposed approach outperforms previous alternatives in most of the studied scenarios

In order to better understand the advantages of our proposal, we performed additional experiments to analyze the learning balance across tasks, network layers, and training steps. The obtained results

provide evidence that a balanced contribution of the different tasks to the network parameter updates throughout the whole training is key for the successful training of multi-task networks. This explains the success of the proposed approach and should be considered for future research in multi-task learning. Additionally, the results herein presented show promising potential for the future application of MAO to different multi-task settings, in computer vision as well as in any other application domain where the optimization of multiple training losses may be required.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work is supported by Ministerio de Ciencia e Innovación, Government of Spain, through the RTI2018-095894-B-I00, PID2019-108435RB-I00, TED2021-131201B-I00, and PDC2022-133132-I00 research projects; Consellería de Cultura, Educación e Universidade, Xunta de Galicia, through Grupos de Referencia Competitiva ref. ED431C 2020/24 and the postdoctoral fellowship ref. ED481B-2022-025. CITIC, Centro de Investigación de Galicia ref. ED431G 2019/01, receives financial support from Consellería de Cultura, Educación e Universidade, Xunta de Galicia, through the ERDF of the European Union (80%) and Secretaría Xeral de Universidades (20%). Funding for open access charge: Universidade da Coruña/CISUG.

References

Almalioglu, Y., Turan, M., Saputra, M. R. U., de Gusmão, P. P., Markham, A., & Trigoni, N. (2022). SelfVIO: Self-supervised deep monocular visual–Inertial odometry and depth estimation. *Neural Networks*, *150*, 119–136. http://dx.doi.org/10.1016/j.neunet.2022.03.005.

Bragman, F. J., Tanno, R., Ourselin, S., Alexander, D. C., & Cardoso, J. (2019). Stochastic filter groups for multi-task CNNs: Learning specialist and generalist convolution kernels. In *Proceedings of the IEEE/CVF international conference on computer vision*.

Caruana, R. (1997). Multitask learning. *Machine Learning*, *28*, 41–75. http://dx.doi.org/10.1023/A:1007379606734.

Chen, Z., Badrinarayanan, V., Lee, C.-Y., & Rabinovich, A. (2018). GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proceedings of the 35th international conference on machine learning*.

Chen, Z., Ngiam, J., Huang, Y., Luong, T., Kretzschmar, H., Chai, Y., et al. (2020). Just pick a sign: Optimizing deep multitask models with gradient sign dropout. In *Advances in neural information processing systems (NeurIPS)*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248–255). IEEE.

Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, *12*(61), 2121–2159.

Eigen, D., & Fergus, R. (2015). Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*.

Gong, T., Lee, T., Stephenson, C., Renduchintala, V., Padhy, S., Ndirango, A., et al. (2019). A comparison of loss weighting strategies for multi task learning in deep neural networks. *IEEE Access*, *7*, 141627–141632. http://dx.doi.org/10.1109/ACCESS.2019.2943604.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Guo, M., Haque, A., Huang, D.-A., Yeung, S., & Fei-Fei, L. (2018). Dynamic task prioritization for multitask learning. In *Proceedings of the European conference on computer vision*.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision*. http://dx.doi.org/10.1109/ICCV.2017.322.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE international conference on computer vision*. http://dx.doi.org/10.1109/ICCV.2015.123.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. http://dx.doi.org/10.1109/CVPR.2016.90.

Hervella, A. S., Rouco, J., Novo, J., & Ortega, M. (2021). Self-supervised multimodal reconstruction pre-training for retinal computer-aided diagnosis. *Expert Systems with Applications*, *185*, http://dx.doi.org/10.1016/j.eswa.2021.115598.

Ilyas, T., Mannan, Z. I., Khan, A., Azam, S., Kim, H., & De Boer, F. (2022). TSFD-Net: Tissue specific feature distillation network for nuclei segmentation and classification. *Neural Networks*, *151*, 1–15. http://dx.doi.org/10.1016/j.neunet.2022.02.020.

Kendall, A., Gal, Y., & Cipolla, R. (2018). Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. CVPR.

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representations*. (ICLR).

Kosti, R., Alvarez, J. M., Recasens, A., & Lapedriza, A. (2017). Emotion recognition in context. In *The IEEE conference on computer vision and pattern recognition*. (CVPR).

Kosti, R., Alvarez, J. M., Recasens, A., & Lapedriza, A. (2020). Context based emotion recognition using EMOTIC dataset. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(11), 2755–2766. http://dx.doi.org/10.1109/TPAMI.2019.2916866.

Liu, S., Johns, E., & Davison, A. J. (2019). End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In *7th international conference on learning representations*.

Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., & Terzopoulos, D. (2021). Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, http://dx.doi.org/10.1109/TPAMI.2021.3059968.

Misra, I., Shrivastava, A., Gupta, A., & Hebert, M. (2016). Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. http://dx.doi.org/10.1109/CVPR.2016.433.

Najar, F., & Bouguila, N. (2022). Emotion recognition: A smoothed Dirichlet multinomial solution. *Engineering Applications of Artificial Intelligence*, *107*, http://dx.doi.org/10.1016/j.engappai.2021.104542.

Nakamura, A. T. M., Grassi, V., & Wolf, D. F. (2021). An effective combination of loss gradients for multi-task learning applied on instance segmentation and depth estimation. *Engineering Applications of Artificial Intelligence*, *100*, http://dx.doi.org/10.1016/j.engappai.2021.104205.

Sener, O., & Koltun, V. (2018). Multi-task learning as multi-objective optimization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in neural information processing systems (NeurIPS), Vol. 31*.

Silberman, N., Hoiem, D., Kohli, P., & Fergus, R. (2012). Indoor segmentation and support inference from RGBD images. In *Proceedings of the European conference on computer vision*.

Standley, T., Zamir, A., Chen, D., Guibas, L., Malik, J., & Savarese, S. (2020). Which tasks should be learned together in multi-task learning? In *Proceedings of the 37th international conference on machine learning*.

Sun, K., Xiao, B., Liu, D., & Wang, J. (2019). Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Tieleman, T., & Hinton, G. (2012). Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.

Vandenhende, S., Georgoulis, S., Van Gansbeke, W., Proesmans, M., Dai, D., & Van Gool, L. (2021). Multi-task learning for dense prediction tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, http://dx.doi.org/10.1109/TPAMI.2021.3054719.

Vandenhende, S., Georgoulis, S., & Van Gool, L. (2020). MTI-net: Multi-scale task interaction networks for multi-task learning. In *Proceedings of the European conference on computer vision*.

Xu, D., Ouyang, W., Wang, X., & Sebe, N. (2018). PAD-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., & Finn, C. (2020). Gradient surgery for multi-task learning. In *Advances in neural information processing systems (NeurIPS)*.

Zhao, Z.-Q., Zheng, P., Xu, S.-T., & Wu, X. (2019). Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*, *30*(11), 3212–3232. http://dx.doi.org/10.1109/TNNLS.2018.2876865.

Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., & Oliva, A. (2014). Learning deep features for scene recognition using places database. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems, vol. 27*. Curran Associates, Inc.