DOCUMENTATION TO ASSIGNMENT 2 - AUXILIARY TASKS

Piotr S. Maciąg
P.Maciag@ii.pw.edu.pl

1 PRELIMINARIES

The following sections contain the solutions to problems 2-5 from the given assignment 2. To create all plots, the *ggplot2* package for the R language is used (Wickham et al., 2018). For the implementation of classification results given in (Du et al., 2018) we used Keras library for R Allaire & Chollet (2019). All source codes (both implementation of classification and generating plots) are available at the Github repository (Maciag, 2019).

2 AD PROBLEM 2

1. The given assignment concerns the problem of solving a *main task* $\mathcal{T}_{main}$ under the assumption that an *auxiliary task* $\mathcal{T}_{aux}$ is given. The authors propose to use auxiliary task $\mathcal{T}_{aux}$ to positively improve a solution of the main task $\mathcal{T}_{main}$, but with the assumption that it will never affect finding the solution of $\mathcal{T}_{main}$ negatively.
2. For each of two task, the loss functions are defined: $\mathcal{L}_{main}$ for the main task and $\mathcal{L}_{aux}$ for the auxiliary task, respectively. Two models $f(\cdot, \theta, \phi_{main})$ and $g(\cdot, \theta, \phi_{aux})$ (e.g. neural networks) are used to solve the main and auxiliary tasks, respectively. The two models share the vector of parameters θ as well as have their separate parameters' vectors (ϕ_{main} for the main task $\mathcal{T}_{main}$ and ϕ_{aux} for the auxiliary task $\mathcal{T}_{aux}$, respectively). In particular, the authors propose to solve $\mathcal{T}_{main}$ by minimization of the overall loss function given in Eq (1).
$$\min_{\lambda^{(t)}} \; \mathcal{L}_{main}(\theta^{(t)} - \alpha \nabla_\theta(\mathcal{L}_{main} + \lambda^{(t)}\mathcal{L}_{aux}) \tag{1}$$
3. To do this end, the authors introduce three propositions and show their usefulness for the proposed solution.
4. The article clearly explains the main problem and the proposed solution. However, in some its parts it could be more detailed:
   - In subsection 3.1 (Experiments on image classification tasks), when comparing different approaches (single-task, multi-task and proposed approach) what are the experimental setups and how single task and multi task are defined.In particular, what is denoted in axis X in Figure 2.
   - Why only cosine similarity is selected for λ approximation. It has been presented in (Kryszkiewicz, 2013) that Tanimoto similarity is strictly related to cosine similarity. Maybe Tanimoto similarity will give better results?
   - The experiments presented in subsection 3.1 are not very much convincing that the proposed solution can give better results that simple single task classification. In fact, the classification for the near pair in nearly the same for the single task classification as for the proposed solution and for the far pair the proposed solution is not giving better results than multi task classification.

3 AD PROBLEM 3

In Figure 1, Figure 7 from (Du et al., 2018) is reproduced. The documented code which generates plots in Figure 1 using the *ggplot2* package is contained in a file entitled *Figure7Plot.R* and included in the github repository (Maciag, 2019).

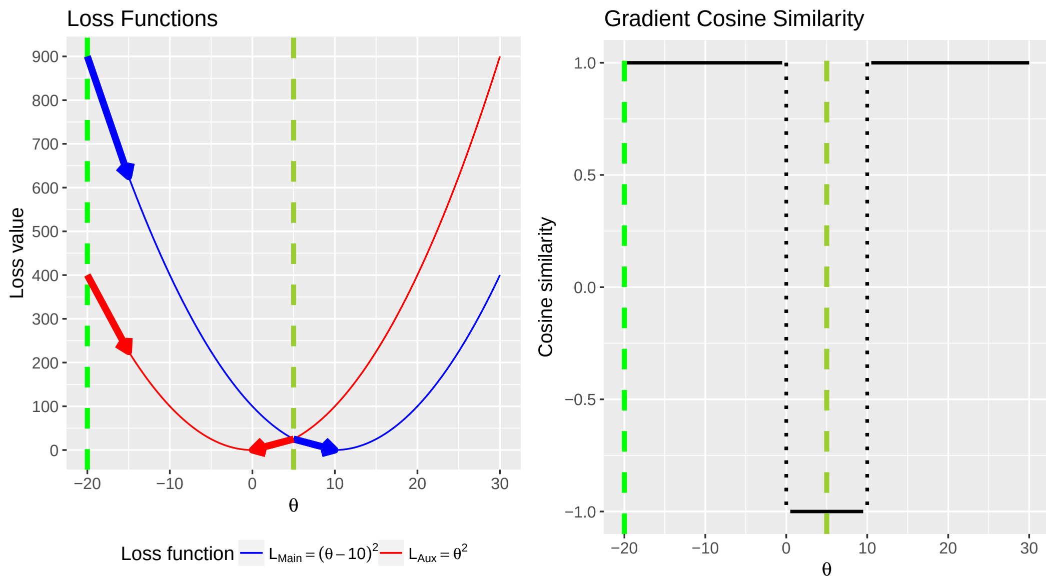

Figure 1: Reproduction of Figure 7 from (Du et al., 2018).

The plots in Figure 1 are generated as follows: first loss and their gradient functions are defined. The loss function for the main task and auxiliary task are plot for all θ between -20 and 30 as presented in *Loss Functions*. Second, for the same range of θ, the cosine similarity function between the two gradient functions is defined and plot as presented in *Gradient Cosine Similarity*.

4 AD PROBLEM 4

In our experiments, we used Keras deep learning library for the R language Allaire & Chollet (2019) with TensorFlow library in version 1.11 (GPU version) Inc. (2019). All computation are performed using GPU.

For the experimental results presented in the next section, we used the following setup of neural network:

- A linear stack of layers is used.
- The first two layers consists of convolution neural networks.
- Two dense layers, with the second one providing the results of classification.

Unfortunately, due to the technical limitations of Keras library (especially when processing large image data as the one used in our experiments) we were not able to implement Algorithm 1 from article Du et al. (2018). In particular, the problem is with obtaining gradients after each batch of data is processed by the `fit_generator` function used to train a neural network. Such gradients can be obtained when the network is trained in the the *eager* mode of TensorFlow library, however the eager mode can not be used with the image processing functions and object (especially `generator` object used to read image data). The possible solution to that problem is to introduce the user own schema of storing image data as tensors. However, time constraints did not allow us to find such solution.

The code in R performing classification is included in file *Classification.R* and contained in the Github repository.

5 AD PROBLEM 5

At the moment of solving the given assignment, the full version of ImageNet dataset used in the experiments in (Du et al., 2018) is under maintenance and is not available for download. Moreover, the full version contains more than 155 GB of data which makes it difficult to use at a typical personal laptop and can not be used as well in the proposed Colab environment.

Because of these reasons, we decided to use a tiny version of ImageNet in our experiments (Fei-Fei et al., 2019). The tiny version of dataset can be downloaded from the same page as the full version and is characterized by the following setup: only 200 classes from the original 1000 classes of ImageNet are selected. The 500 images for each one from the 200 classes are given. The resolution of each image is 64 x 64 pixels and 3 channels of colors RGB are given. Unfortunately, not all categories originally used in the experiments in Du et al. (2018) are available in the used tiny version of ImageNet. Because of that, we selected three pairs of near categories and three pairs of far categories as follows:

- For near categories: Egyptian cat and Persian cat; Golden retriever and Labrador retriever; Monarch butterfly and Sulphur butterfly.
- For far categories: elephant and teapot; baber shop and lemon; chimpanzee and orange.

All other images contained in a tiny ImageNet are used as the *background* images in classification.

To compare classification results for the multi task (where each from selected two classes in a pair is classified against the other one and against the background images) and single task (where only the first class from a pair is classified against the set of background images), we choose: from the near pairs Egyptian cat vs. Persian cat and from the far pairs elephant vs. teapot. The results of classification are provided in Figure 2. Figure 2 is generated using *Rplots.R* file contained in the Github repository.

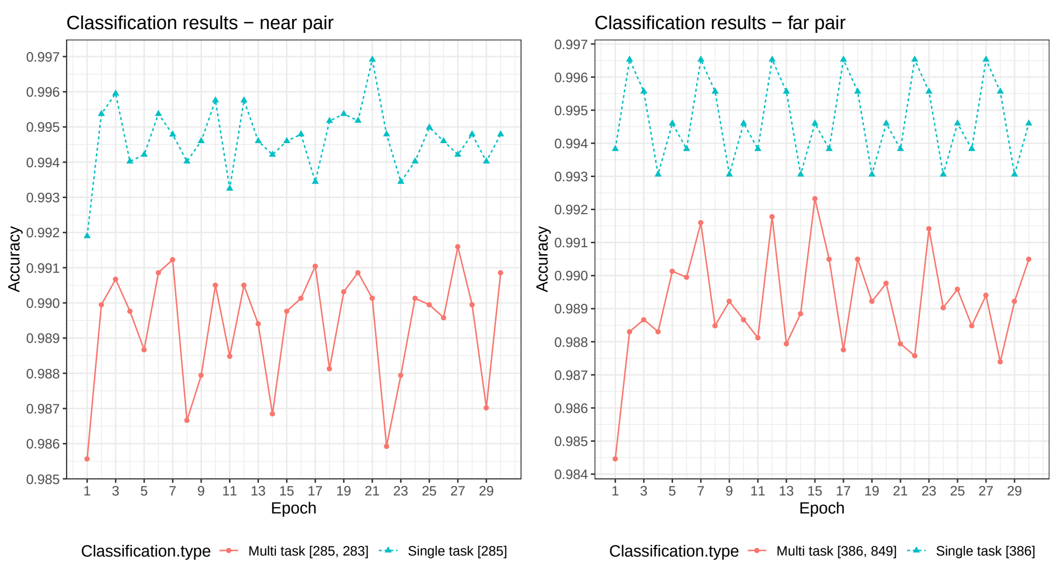

Figure 2: Reproduction of classification results for two types of images: for the near pair we selected Egyptian cat (class id 285) and Persian cat (class id 283). For the far pair African elephant (class id 386) and teapot (class id 849) are selected. The plots are illustrating two classification task: multi task where each selected category is classified against all other images (background) and single task where only the first selected class from each pair is classified against the set of background images.

REFERENCES

JJ Allaire and François Chollet. Keras library for the r language, 2019. URL `https://keras.rstudio.com/`.

Yunshu Du, Wojciech M Czarnecki, Siddhant M Jayakumar, Razvan Pascanu, and Balaji Lakshminarayanan. Adapting auxiliary losses using gradient similarity. *arXiv preprint arXiv:1812.02224*, 2018.

Li Fei-Fei, Kai Li, Olga Russakovsky, Jonathan Krause, Jia Deng, and Alex Berg. Tiny imagenet, 2019. URL `https://tiny-imagenet.herokuapp.com/`.

Google Inc. R interface to tensorflow, 2019. URL `https://tensorflow.rstudio.com/`.

Marzena Kryszkiewicz. On cosine and tanimoto near duplicates search among vectors with domains consisting of zero, a positive number and a negative number. In Henrik Legind Larsen, Maria J. Martin-Bautista, María Amparo Vila, Troels Andreasen, and Henning Christiansen (eds.), *Flexible Query Answering Systems*, pp. 531–542, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-40769-7.

Piotr Maciag. Github repository with the code for the assignment, 2019. URL `https://github.com/piotrMaciag32/Tooploox---task-2`.

Hadley Wickham, Winston Chang, Lionel Henry, Thomas Lin Pedersen, Kohske Takahashi, Claus Wilke, and Kara Woo. ggplot2 r package: Create elegant data visualisations using the grammar of graphics, 2018. URL `https://cran.r-project.org/web/packages/ggplot2/ggplot2.pdf`.