Soft labels for classification on uncertainty *prodiplosis*

Gabriel Quinche[*]

11 de octubre de 2022

Resumen

Neural network classification of object has a long story, from the classic digit classification problem to transformers networks, we have evolved by switching from more classic components like the arc tangent function matrix multiplications and euclidean losses, to (leakey) Relus, batch normalization layers, to soft cross entropy. The object of this document is to showcase some advantages of using that last element, also called Cross entropy with soft labels, its mathematically and equivalent implementation in the library *pytorch* is the KL divergence, a measure connected to the transformative information theory area, this measure let us compare probability measures, armed with this tool we can teach in more robust ways our classification networks, by evading some problems related to the non convexity of the euclidean difference between vectors and tackling it with a tool related specifically to probabilities (0 to 1 range) of belonging or not to a number of classes. We will show some comparisons to the more traditional log loss, which takes only into account targets that are binary *one* or *zero* and which we suggest suffers from a loss of information. To showcase this we will present two problems related to computationally agronomy, classification of insects and the end estimation of a ratio of sick leafs vs healthy ones. With the first one we will show a mathematical comparison of agreement scores with soft labels vs classical ones, taking advantage of more modern *small data* techniques. We will conclude by highlighting why this technique will be useful precisely in problems of this area, where some classification problems are even difficult for a human expert and a probability of belonging to a class is a preferred way to represent uncertainty.

1. Introduction

Neural networks, and in general machine learning models learn to classify or predict by the use of a set of training examples. Most popular methods use a technique were this training example

[*]UNAL facultad de Ciencias

already have the correct label, or value to be predicted, usually by doing some previous work with experts collecting carefully samples. However in problems where expert knowledge is still in is infancy, for example classification of new unknown species, we shouldn't trust so blindly in the labels we assign, more so, as in digital agronomy this tools could be used by regular people trying to build a business, grow a new crop or even just feed their family, the efficacy of our tool should be pushed as much as it could be done: its crucial to do our best to create models that predict the truth as close as possible (aside of doing without important side effects) but also reflect our uncertainty in some positions.

2. Mathematicall background

2.1. KL and crossentropy relation

3. Procedure

With classic machine learning training techniques if we were to use log loss and a set of soft labels, our neural network wouldn't converge, as log loss is not designed to give useful information for intermediary class belonging (for example a 70 % positively predicted class). We learned this truth the hard way, after many many attempts tweaks and architectural changes to our network, for the contrary the initial, and somewhat naive use of euclidean loss, was a first working prototype that we were failing to reproduce with the more widely suggested log loss, although not recommended specially when training more classical models like logistic classifiers, euclidean loss does give useful information to a model so that i tries to learn not only categorical but uncertainty in classes. Before arriving at a final choice with the KDL loss implemented in Pytorch other interesting alternatives came to mind when we realized that the problem was our choice of soft labels: [1] even when using hard labels we could train the network to try to learn the percentage as a multi class problem, as such we could simply have several classes representing intervals of certainty, for example having four classes representing $\{(0, 25\,\%), [25\,\%, 50\,\%), [50\,\%, 75\,\%), [75\,\%, 100\,\%)\}$, moreover as to preserve a little more granularity and by using some loss measures already in pytorch like multi label losses, we could transform it also to a problem of bisecting the $(0, 1)$ interval, where with the classic yes or no question related to the entropy number, we could achieve for example a precision or length of interval of $\pm 0{,}097\,\%$ [2]

[1] This was a choice that however seemed like the natural one as percentages not only reflected better what our -still learning- expert could guess about the belonging to a class, but also that in the sense of information theory (or even strict type bit length of bools vs floats), percentages preserved more information

[2] The reason for this is that with 10 yes and no question we have 2^10 possible options, then by diving 1 in 2^{10} equally spaced intervals, we get an interval length of 0,097 %, a possible interesting question is the practicality of this or other interval distributions and comparing it to the multi class classification problem, the granularity of this method however could be a good indicator of how multi label classification has to be a much more complex problem than multi class classification

3.1. Technical specifications

Here we should add the tech specs of the cpus used to run the experiment and estimated times for them

4. Comparison

two comparison are suggested to see the advantages of the pytorch implemented KDL. [3] as we have so little amount of data, and acquisition of more would be cost prohibitively we would use another novel technique we would like to explore a technique related to the paper of [1] where comparing agreement on untrained example we try to predict model performance on the real world, as such to measures are to be taken, first a simple standart deviation between predicted certainties, and second a mean of KDl pairwise divergence, a sample where we would compute both of these measures would look like $f_i(\text{testImage}_1)$ for every i representing a different k [4] fold cross validated model,finally by using a set of around 10 to 20 images of insects we would have a cloud of points for either the classical technique or the KDL divergence approach, we can add some extra cloud points by selecting different cutoff points for the classical approach, and finally with a set of around 100 datapoints (20 for each category) we would make two anovas test where we would hopefully show that KDL method has a significant impact on agreement in both pairwise measures and standart deviation.

The approach of using agreement would help us evade problems related to leakeage and also to diverge from the classical accuracy competition, where models attain 90 % to 99 % accuracy or even f_1 without highlighting the poor results on field that this measures cant account for.

4.1. Selected k parameter

4.2. Anova test

5. Conclusions

link

[3]which can be accessed with cross entropy loss

[4]we would like to choose an arbitrary but constant k related specifically to the species of study *prodiplosis*, where $k = 11$ is the number of characters in its short name, for future experiment a more elaboratore decision on k could be made, whereras problems like leakeage, normality of scores and others are taken into account

Referencias

[1] C. Baek, Y. Jiang, A. Raghunathan, and Z. Kolter, “Agreement-on-the-line: Predicting the performance of neural networks under distribution shift,” *arXiv preprint arXiv:2206.13089*, 2022.

[2] J. Cabrera and E. Villanueva, “Investigating generative neural-network models for building pest insect detectors in sticky trap images for the peruvian horticulture,” in *Annual International Conference on Information Management and Big Data*, pp. 356–369, Springer, 2022.

[3] Y. Bengio, “Practical recommendations for gradient-based training of deep architectures,” in *Neural networks: Tricks of the trade*, pp. 437–478, Springer, 2012.