The Computer Science Department

The Technion-Israel Institute of Technology

RepMet: Representative-based metric learning for classification and few-shot object detection

This is an undergraduate project in

the Geometric Processing Laboratory

By: Nir Shopen and Omer Kawaz

Supervisor: Alona Goltz

Date: 22/01/2022

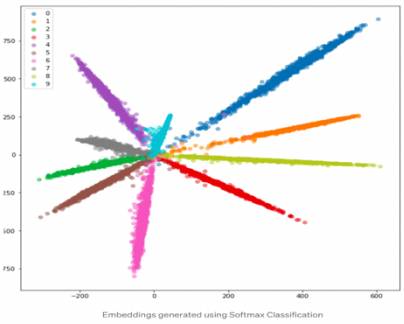
Introduction

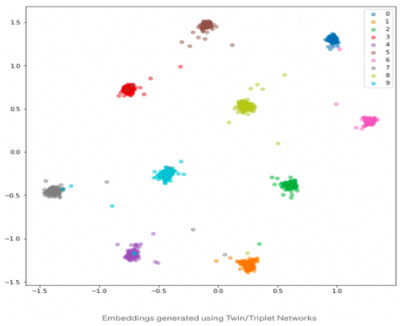
The aim of the project was to implement a DML classification network according to the article RepMet: Representative-based metric learning for classification and few-shot object detection.
The article describes the system architecture we relied on when we used the Python language and backbone network provided to us by Alona.
After implementing the architecture and optimizing the hyper parameters the network provided results that are not necessarily better than the base version.
Better results may be obtained by expanding the amount of hyper-parameters and making an informed choice of augmentations.

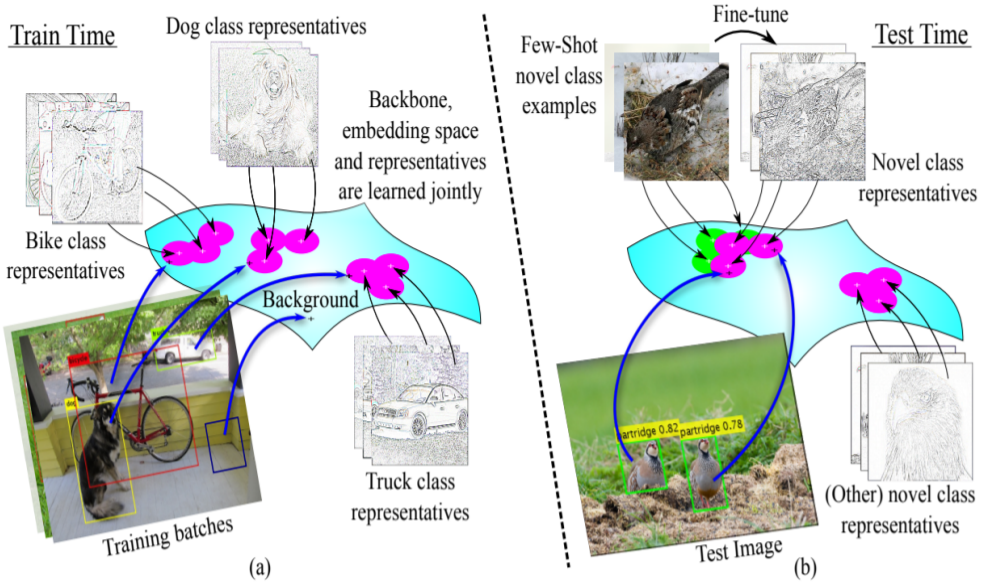
Background

The approach taken by the article is DML - Distance Metric Learning.
A method in which each image and representative is represented by a vector in space (of a predetermined size) and the purpose of the network is to produce representations so that vectors that represent objects from identical classes are closer in space than vectors that represent objects from different classes.

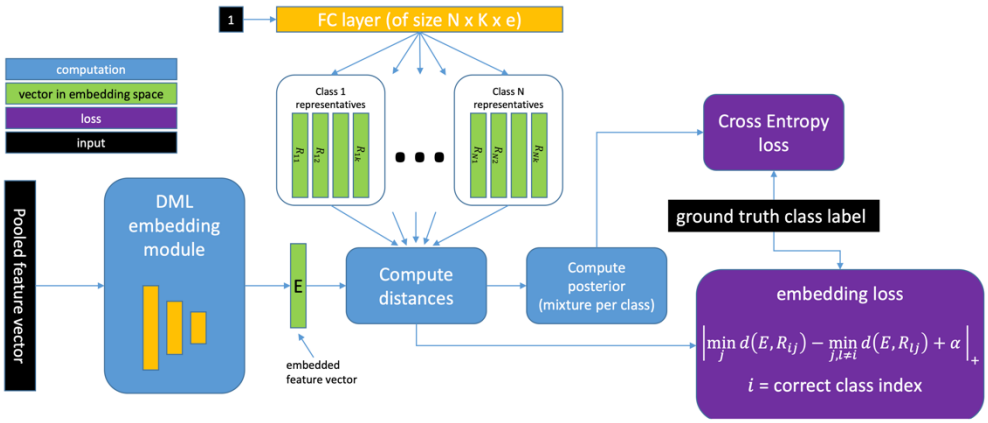
This aspect of learning discriminative features is what metric learning achieves.







System description



The system consists of the following parts:

Backbone network, DML embedding module, FC Layer layer, two loss functions (Cross-entropy loss and Embedding loss), and a posterior calculation system.

The backbone network was provided to us by Alona and the cross-entropy loss function is given in the pytorch library.

The DML embedding module was implemented as required in the article using two FC Layers, batch normalization and ReLU non-linearity.

The FC Layer layer is initialized with Scalar 1 and its size is N * K * E when:

N number of departments

K number of representatives

E Size of the vector representing the space.

After calculating the distances between the representatives and the output collector, the results were transferred to the loss functions and the posterior calculations (percentages of affiliation to the department).

The embedding loss function is implemented by us according to the instructions of the article and depends on the alpha parameter

The posterior calculations were realized by us as well and depend on the sigma parameter (choice between equations 2 and 5 according to performance).

The results are then forwarded and all parts of the system train together and a new epoch begins.

Description of performance

The system architecture depends on many parameters as well as posterior calculations that depend on the choice between two equations (2 and 5).

The algorithm on which the system is built is a rigid algorithm (except for the choice between equations 2 and 5) which means we have no option to test a different implementation or change the structure of the system. The way to bring the system to maximum efficiency is by optimizing hyper parameters.

We selected a number of hyper parameters that we thought were the most important and on which we ran optimization using a serial method.

We ran for a start, for each of the hyper parameters, a test to determine the tuning range by running the whole system with default values and a wide range for the parameter being tested and a review of the system results for each value.

After finding the initial optimal ranges we ran the system on all the parameter ranges we found until we reached optimal results.

Hyper parameters

We selected the hyper parameters according to the article and according to the general parameters of the training

The parameters we chose are:

Alpha (appears in equation number 4, the embedding loss equation),

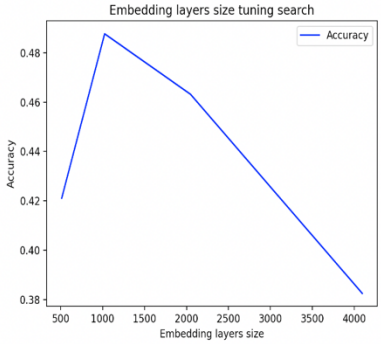
Sigma (appears in equation number 1 and 2 and 5 contain it),

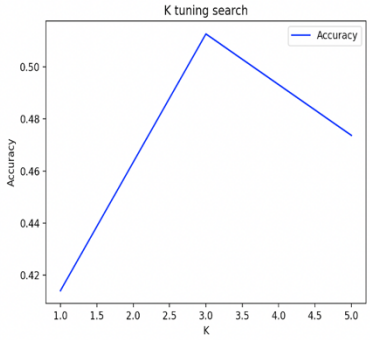
K (base number of representatives)

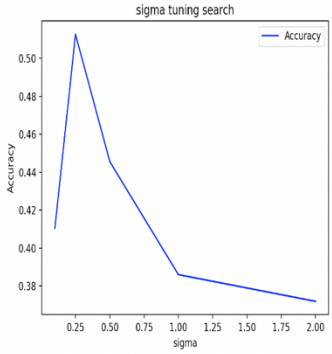
Learning rate,

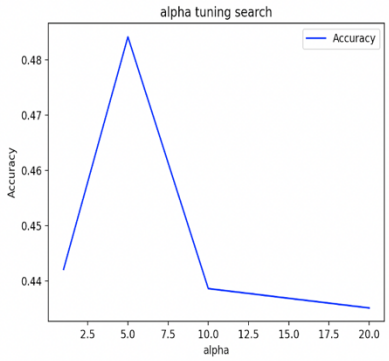
And the size of the embedding layers.

Initial tuning graphs





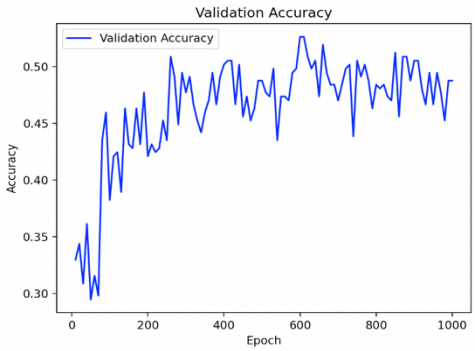




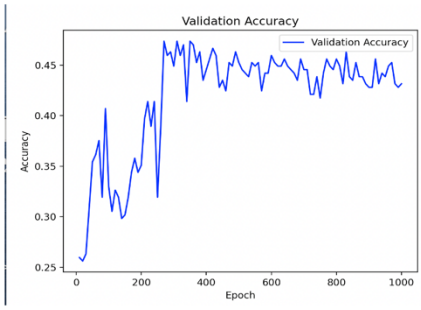
Results

After all the runs we found the values of the ideal parameters K = 3, sigma = 0.25, alpha = 1, learning rate = 0.0001 embedding layers = 1024 ,.

The following are the results of the algorithm (with equation 5, as recommended by the authors of the article) with the above values versus the baseline results:



baseline



Best parameters

It can be seen that the algorithm results are not necessarily better than the baseline results and that its performance is more or less fixed after about 300 epochs.

In addition, we also chose to examine equation number 2 instead of equation number 5, contrary to the recommendation of the authors of the article, understanding that each data set has a slightly different architecture, and we also found that better results were obtained:

	accuracy	precision	recall	F1	AP
Baseline	0.5263	0.5331	0.5053	0.5109	0.5590
Optimal(2)	0.5127	0.4710	0.5127	0.4620	0.4720
Optimal(5)	0.4841	0.4533	0.4841	0.4128	0.4237

Conclusions and recommendations

The quantity and range of selected hyper parameters are insufficient for the current data to obtain a network that provides priority results.

There are many additional parameters, and we do not rule out that turning them into hyper parameters and adjusting them may provide even better results.

For example: the vector size of the representatives in space (size E in the article)

In addition, as reviewed on the results page we saw that equation 2 was superior to equation number 5 although in the article the authors state that equation number 5 is preferable for classification purposes. Therefore, according to the results we do not think that the conclusion of the authors of the article can be relied upon and both equation number 2 and 5 should be examined when looking for the ideal architecture.

References

RepMet: Representative-based metric learning for classification - and few-shot object detection
https://arxiv.org/pdf/1806.04728.pdf

By: Leonid Karlinsky∗, Joseph Shtok∗, Sivan Harary∗, Eli Schwartz∗, Amit Aides, Rogerio Feris IBM Research AI