A REPORT ON

PROTOTYPICAL NETWORKS

submitted in partial fulfilment of the requirements of the course CP302 Capstone Project- I for the award of the degree of

Bachelor of Technology
in
Mathematics and Computing
by

Rishi Johri (2019MCB1287), Anirudh Sanghi (2019MCB1237)

under the guidance of

Prof. Arun Kumar and Prof. Deepti Bathula

DEPARTMENT OF MATHEMATICS
INDIAN INSTITUTE OF TECHNOLOGY ROPAR
RUPNAGAR-140001
PUNJAB - INDIA
May 2022

Certificate

This is to certify that Mr. Rishi Johri and Mr. Anirudh Sanghi, students of B. Tech. (Mathematics & Computing) has worked on the Project titled “Prototypical Networks”, under the course CP302 Capstone Project- I, under my supervision and guidance.
May 22

Signature of the Guide
Prof. Arun Kumar
Associate Professor
Department of Mathematics
I.I.T. Ropar
Rupnagar- 140 001
Punjab - India.

Signature of the Co-Guide
Prof. Deepti Bathula
Associate Professor
Department of Computer Science and Engineering
I.I.T. Ropar
Rupnagar- 140 001
Punjab - India.

Acknowledgement

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

We are highly indebted to Prof. Arun Kumar and Prof. Deepti Bathula for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

We would like to express our gratitude towards the Department of Mathematics and the Department of Computer Science for their kind co-operation and encouragement which has helped us complete this project.

Our thanks and appreciations also go to our colleagues in developing the project and people who have willingly helped us out with their abilities.

Abstract

In recent times, different deep learning techniques have shown excellent results in predicting several classification tasks. Though, all those techniques rely on the fact that there is an ample amount of data for training. Meta-Learning is a technique that targets how to learn when there is less data. Few-Shot learning is one of the types of meta-learning in which there can be multiple approaches eg. - Transfer learning, prototypical networks etc. We work upon the approach of the prototypical network where a classifier learns to generalize to new classes such that they are not seen in training examples and be able to classify new classes with only seeing very few examples of them. Prototypical networks learn a latent space and identify the prototypical representation of each class in that space to classify the data points by computing distances from chosen prototypes. We have produced the baseline results on the dermatological dataset using the original prototypical network algorithm. We used several different methods like using non-euclidean metrics and altering the loss function to maximize the distance between prototypes and using medoids to find the prototypes. We observe a considerable increase in the accuracy of 5-8 % from baseline results.

Prototypical Networks

Rishi Johri(2019MCB1287), Anirudh Sanghi(2019MCB1237))

Contents

1 Introduction

Over the last few years, many deep learning algorithms and corresponding models have been developed in the computer vision domain. Usually, these models are very complex and consist of millions of trainable parameters. These models are capable of doing classification tasks and achieving accuracies as high as 99.9%. But all these algorithms require large annotated datasets to learn a good set of parameters and give considerably good results. However, in the real world, there are many problems where we have access to very few examples to get our model trained. Getting more data in these cases might be simply infeasible or very expensive viz. Medical datasets. To solve this problem, meta-learning techniques are used.

Meta-learning also known as ‘’learning to learn” is a machine learning approach which targets how to learn rather than what to learn which standard machine learning algorithms do. Few-shot Learning is a kind of meta-learning in which the classifier has to adapt to accommodate a set of classes that are not present during training, giving very few examples of these classes. In the naive approach, one can retrain our model on a new dataset multiple times but it would not solve the problem as it will cause overfitting of the data.

In Few-shot learning, we sample different tasks which are exact replicas of actual few-shot learning problems(or tasks) i.e. learn on a few examples and then classify unseen data points. It trains the model on multiple such tasks where each task is independent of the other. For each task, the model first learns on a subset of the training dataset called as ‘support set’ and then it evaluates the model on another subset of the training dataset called as ‘query set’ to update the model parameters. Many different approaches to solve the Few Shot Learning task have been proposed in the recent years such as Matching Networks proposed by Vinyals et al.[2], A Meta-Learning Approach to Few-shot Learning proposed by Ravi and Larochelle[3], Prototypical Networks for few-shot learning by Jake Snell[1].

We have used the approach of the prototypical network on the dermatological dataset - Derm7pt to produce the baseline results. Further, we performed various modifications to the algorithm in order to obtain better results. Prototypical Networks are based on the idea of learning the latent space which could represent the dataset it is trained on and accommodate any unknown data it encounters. The latent space is learned such that data points belonging to the same class form cluster. Each cluster formed can simply be represented by a representative point (prototype).

Our first approach toward modifying Prototypical Network was to try out different latent spaces for the prototypical network. To achieve this we tried different Norms for the latent space, including but not limited to the L1 norm, L3 norm, fractional norms etc. In higher dimensions and in non-euclidean spaces the concept of the centroid is not a good representation of the representative point of a cluster. Therefore in our second approach, we used Medoids (instead of centroids) to represent the representative point of the cluster.

We also observed that mathematically the loss function tries to concentrate individual points of each cluster towards their representative point. We added another parameter to the loss function which would encourage the model to also separate out individual prototypes from each other.

Lastly, we performed different hyper-parameter tuning on the model and tried various combinations of the above-mentioned approaches.

2 Preliminaries

2.1 Prototypical Network

2.1.1 Introduction

Prototypical Networks are based on the idea of learning the latent space which could represent the data set it is trained on and accommodate any unknown data it encounters. We use neural network in non linearly mapping the input to the embedding space or latent space. The latent space is learned such that data points belonging to the same class form cluster. Each cluster formed can simply be represented by a representative point (prototype) that can be calculated by mean of the support set in the latent space.Classification is then performed for an embedded query point by simply finding the nearest class prototype.The model works on the assumption that there exists an embedding space which can form separate clusters of all the classes in the data set. The algorithm of prototypical network is simple, intuitive and very easy to understand.

2.1.2 Definitions

Support Set (S)

In few shot classification we have support set containing N annotated data points represented as $S = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$ where each $x_i \in R^D$ where D is represents D-dimensional embedded space with $y_i \in \{1, 2, ..., K\}$ (where K is number of classes) as corresponding labels. S_k represents the set of data-points whose corresponding labels are k and $S_k \subseteq S$.

Prototypical Network (f_ϕ)

Prototypical networks map each data point into M dimensional latent space as $f_\phi : R^D \rightarrow R^M$ where ϕ are learnable parameters of neural network.
For our Experiments and creating a Baseline Prototypical Network, we have considered a Prototypical network as a Convolutional Network consisting of 6 convolutional blocks. Each convolution block consists of a convolution layer, a Batch Normalization Layer, A ReLu Activation Layer and a max-pooling layer. This is done to match the architecture of Prototypical Network as mentioned by Mahajan in Meta-Derm Diagnosis.

Prototype (c_k)

For each class k its corresponding prototype c_k can be calculated as mean vector of embedded data points of support data points of the corresponding class.

$$c_k = \frac{1}{|S_k|}\Big(\sum_{(x_i,y_i)\in S_k} f_\phi(x_i)\Big) \tag{1}$$

Query Set (Q)

In few shot classification we have support set containing P annotated data points represented as $Q = \{(x_1, y_1), (x_2, y_2), ..., (x_P, y_P)\}$ where each $x_i \in R^D$ where D is represents D-dimensional embedded space with $y_i \in \{y_a \big| (x, y_a) \in S\}$ (y_i comes from same set of classes as the ones present in Support Set) as corresponding labels. Q_k represents the set of data-points whose corresponding labels are k and $Q_k \subseteq Q$.

Task

In few shot learning, the task can simply be defined as correctly classifying datapoints from query set Q given the support set S. If there are N_c classes in the Support set and $|S_k|$=N_s for any class k in S, then this task is commonly called N_c **way N_s shot classification Task**

Now given a distance function $d : R^M \times R^M \rightarrow R^+$ for each query point x using Softmax over distances to the prototypes we get a probability distribution as:

$$p_\phi(y = k|x) = \frac{exp(-d(f_\phi(x_i), c_k))}{\sum_{k'} exp(-d(f_\phi(x_i), c_{k'})}$$

Further, ϕ parameters are learned by minimizing the loss function $J(\phi) = -log(p_\phi(y = k|x))$ of true class with SGD (Stochastic Gradient Descent) . Now different task are sampled from the training data set by randomly choosing a subset of classes. After than from each class choosing a set of examples as support set and remaining as query data points. Following is the pseudo code for the algorithm [cite paper].

Accuracy

Let us sample N tasks from the dataset D. Each task T_i can be represented by combination of a support set to form the prototypes and a query set to evaluate the prototypical networks. Each example in the query set is assigned to the class of the nearest prototype. $A_i$represents the accuracy of correctly classifying the query datapoints to their respective classes for a Task T_i then the accuracy A of the Prototypical Network can represented as

$$A = \frac{\sum_{i=1}^{N} A_i}{N}$$

Algorithm : Computing training loss for prototypical network given N data points and K number of classes in training data set. N_S and N_K are number of support and query examples in each class. RANDOMSAMPLE(S,N) samples N data points from a set S uniformly without replacement.

```
Input: Training data set $D = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$ with $y_i \in \{1, 2, ..., K\}$
. $D_k$ is subset of D such that $\forall (x_i, y_i) \in D_k, y_i = k$. $N_C$ number of class per episode
Output:
    $V \leftarrow$ RANDOMSAMPLE($\{12, ..., K\}, N_C$)
    for k in $(1, 2, ..., N_C)$ do
        $S_k \leftarrow RANDOMSAMPLE(D_{V_k}, N_S)$
        $Q_k \leftarrow RANDOMSAMPLE(D_{V_k} \backslash S_k, N_Q)$
        $c_k \leftarrow \frac{1}{|N_C|} \Big( \sum_{(x_i, y_i) \in S_k} f_\phi(x_i) \Big)$
    end for
    $J \leftarrow 0$
    for k in $(1, 2, ..., N_C)$ do
        for $(x, y)$ in $Q_k$ do
            $J \leftarrow J + \frac{1}{N_C N_Q} [d(f_\phi(x), c_k) + log \sum_{k'} exp(-d(f_\phi(x), c_k))]$
        end for
    end for
```

2.2 Dataset

In our experimentation we have used a dermatological data set - *Derm7pt* . Prototypical network was first tested on *Derm7pt* in Meta-Derm Diagnosis by Mahajan. In the paper they performed experiments on 2 way 3 shot scenario and 2 way 5 shot scenario. They have also stated that to produce considerable results they have performed various augmentation operations on the dataset. However the exact details of the augmentation are not provided in the paper. Therefore we cannot directly compare our results with those mentioned in Mahajan's paper[7].
In our experimentation, we have used the original prototypical network algorithm along with the hyper parameters used in Mahajan's paper[7].
To ensure consistency of results we have performed 5 folds Validation on the dataset. In each fold we keep 13 classes in the training set and 6 classes in the test set. Each class in training dataset contains around 94-95 images while each class of of testing dataset contains around 13-15 images.

2.3 Results

Type of Exp.	2 way 3 shot	2 way 5 shot	5 way 3 shot	5 way 5 shot
Baseline results	75.6	78.03	44.99	48.49

Table 1: baseline Accuracies according to Model Architecture mentioned by Mahajan

All of the above accuracies show a variance of of 1-3%, therefore result of further studies can be considered significant only if the the difference result is greater than 3%. All the results in the subsequent sections would be compared with the baseline results mentioned in table 1.

2.4 Inferences

From the table we can observe that if we keep number of ways constant ,5 shot gives better accuracy then 3 shot, which can be understood as we are providing more data per class in 5 shot to make the model learn. Further 2 way accuracy is better than 5-way accuracy which can be understood as in 2 way there are only two classes, so even without training base accuracy would be 50% whereas in 5 way base accuracy would be 20%.

3 Effect of Different Distance Metrics on Prototypical Networks

3.1 Introduction

The choice of distance metric is a very important factor when it comes to learning the embedding space in Prototypical Networks. Since Prototypical Networks works on the assumption that the dataset can be mapped to the embedding space, it is necessary to choose a embedding space which can efficiently form clusters of different classes in the dataset. For eg., the original paper on Prototypical networks mentions that the Euclidean Distance metric greatly outperforms Cosine similarity when applying Prototypical Networks. However, we noticed that not much experimentation is done to check the effects of different distance metrics on the output of the Prototypical Network.

In higher dimensions the euclidean distances seem to concentrate; all distances between pairs of data elements seem to be very similar. Therefore, the relevance of the euclidean distance has been questioned in the past, and fractional norms (Minkowski-like norms with an exponent less than one) were introduced to fight the concentration phenomenon[4].

Therefore for the purpose of our study, we restricted ourselves to studying the effects of different p-norms on the output of the Prototypical Network. We compare outputs of the prototypical network in terms of the accuracy of the Prototypical Network on the Test Dataset.

3.2 Definitions

3.2.1 p-Norm

Let p be a real number ($p \geq 1$). The p-norm (also called lp norm or Minkowski Norm) of vector x=($x_1, x_2, ...x_n$) is defined as

$$||x||_p = \left(\sum_{i=1}^{n} {x_i}^p \right)^{1/p}$$

For $p = \infty$ we get the max norm which can be defined as

$$||x||_\infty = max_i(x_i)$$

3.2.2 Fractional Norm

In the case of p-norm if $0 < p < 1$ then the resulting distance metric cannot be classified as a norm, since it would violate the triangle inequality. However fractional distance metrics is still a widely used distance metric when it comes to finding the nearest neighbour in various data analytic task.

Mathematically, let p be a real number ($p \leq 1$). The p-norm (also called lp norm or Minkowski Norm) of vector x=$(x_1, x_2, ...x_n)$ is defined as

$$||x||_p = \left(\sum_{i=1}^{n} {x_i}^p\right)^{1/p}$$

3.3 Results

Type of Exp.	p=1	**p=2**	p=5	p=6
2 way 3 shot	68.72	**75.6**	75.33	74.65
2 way 5 shot	75.17	**78.03**	77.61	76.56
5 way 3 shot	34.93	**44.99**	43.32	44.47
5 way 5 shot	44.34	**48.49**	47.73	47.15

Table 2: Accuracies with different p values in Minkowski norm

All the accuracies mention above are average of experiments. It was observed that the accuracy can vary upto 5%. We hypothesize that this deviation happens due to randomness of creating tasks from the dataset

3.4 Inferences

From the above results, it is evident that euclidean norm performs better than any other Minkowski norm tried in the experiments. **Note**: Due to nature of training, and calculating distances in higher dimensions, calculating fractional norms results in negative exponent of distance to converge to 0. This would in turn prevent the loss function to properly update the parameters. Therefore it can be concluded that applying fraction norm is infeasible in current architecture of Prototypical Networks

4 Using Different Clustering Approach to form Prototype

4.1 Introduction

When we analyze the distance metric in higher dimensions we find a lot of properties which are generally not intuitive. For eg. let us take a hyper-cube in higher dimensions (d dimensions). one of the corner of the hyper cube is at the origin. The edges of the cube extend for 1 unit in the positive direction of the axes. if we take random variables $x = [x_1, x_2, ...x_d]$ and $y = [y_1, y_2, y_3.....y_d]$ where each x_i, y_i comes from a uniform distribution from 0 to 1. In this case the Euclidean distance between x and y will be

$$d = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}$$

Since d is large, we can expect that for some i, $(x_i - y_i)$ will be close to 1. That puts a lower bound of 1 on the distance between almost any two random points. In fact, a more careful argument can put a stronger lower bound on the distance between all but a vanishingly small fraction of the pairs of points. However, the maximum distance between two points is $\sqrt{d}$, and one can argue that all but a vanishingly small fraction of the pairs do not have a distance close to this upper limit. In fact, almost all points will have a distance close to the average distance [5]. This suggests that there is almost no pair of points that are close to each other. This would in turn mean that there are no grounds to form clusters (through centroids) for any group of points in Euclidean space.

Prototypical networks do not enforce any rules on density of the clusters, therefore for the sake of simplicity, we can assume that probability that the clusters are of uniform density is maximum. If the clusters are of uniform density then the centroid obtained would not correctly represent the cluster since any other point in the cluster space would also give almost the same sum of distances from the datapoints. To solve this problem we decided to use medoids instead of centroids to represent the cluster. The idea is that since the Prototypical Network learns the mapping of data point to the embedding space, using medoids would encourage the network to form clusters which are not of uniform density.

4.2 Definitions

4.2.1 Prototype

A representative point of the cluster is known as the prototype. There are many ways to define the representative point but for the purpose of our study, for points $x_1, x_2, ...x_n$ belonging to a particular class, their prototype p aims to minimize D. Where D is defined as

$$D = \sum_{i=1}^{n} d(p, x_i)$$

4.2.2 Centroid

For points $x_1, x_2, ...x_n$ belonging to a particular class, their centroid c is defined as

$$c = \frac{\sum_{i=1}^{n} x_i}{n}$$

It is important to note that for non euclidean spaces, the centroid does not minimize D.

4.2.3 Medoid

For points $\{x_1, x_2, ...x_n\}$ belonging to a particular class, their centroid m is defined as

$$m = argmax_{a \in \{x_1, x_2, ...x_n\}}(\sum_{i=1}^{n} d(a, x_i))$$

4.3 Results

Type of Exp.	p=1	**p=2**	p=5	p=6
2 way 3 shot	71.03	**74.42**	74.02	73.69
2 way 5 shot	74.05	**74.88**	75.39	74.56
5 way 3 shot	43.17	**45.69**	42.98	43.24
5 way 5 shot	44.75	**42.32**	45.66	46.83

Table 3: Accuracies when using Medoid of clusters as the prototype, where p is the hyperparameter corresponding to different Minkowski norms

4.4 Inferences

Comparing the acccuracies in Table 3 with those in Table 2 shows that clustering through centroid is a better approach than clustering through medoids. We infer that this happens because Medoid converges towards true representation (prototype which can perfectly classify all query data-points) of cluster with increase in data points. However in few shot learning the algorithm does not have many datapoints to iterate through.

5 Loss function Alteration

5.1 Introduction

In prototypical network, neural networks are used to map input images to the embedding space and based on the prototypes formed from support set. After that embedded query points are evaluated to compute the loss function as $J(\phi) = -log(p_\phi(y = k|x))$, where k is true label of x. Now, here loss function is maximizing the $p_\phi(y = k|x)$ i.e. maximizing the softmax probability of a data point of being in its true class cluster. Further, from equation 2, we can observe that

$$p_\phi(y = k|x) \propto \frac{1}{d(f_\phi(x), c_k)}$$

So, by maximizing the $p_\phi(y = k|x)$, loss function is minimizing the distance of embedded data point x from the cluster prototype c_k, that ultimately aims to make each cluster spread minimum in the latent space. Now, we propose the hypothesis that during testing, if different clusters are close to each other even though within cluster spread is minimum, then the confidence level of the prediction will be low because there is not significant difference between the distance from predicted class prototype and other class prototypes.

So, now we alter the loss function such that it include a penalizing term that will penalize the inter-cluster distance for each cluster from the nearest cluster prototype such that less the distance between the clusters more the penalizing term magnitude. This technique gave us considerable improvement in accuracy from the baseline results.

Since the new loss function would try to separate out the prototypes we expected a rise in both training and testing accuracies. This may give rise to overfitting of the dataset. Therefore for the experiments in this section we have restricted the number of convolution blocks in the Prototypical Network to 4 instead of 6. This would make the model simpler and thus prevent overfitting of training data.

5.2 Definitions

5.2.1 Nearest Prototype c_i^*

We define a term c_i^* where $i \in (1, 2, ..., N_c)$ that represents a prototype whose distance is minimum from the prototype c_i.

$$c_i^* = \arg\min_{c_j, i \neq j} d(c_i, c_j)$$

5.2.2 CSF(Penalty Term)

We now define a term called as Cluster Separator Factor(CSF) that will penalise the distance between the a prototypes and its corresponding nearest prototypes.

$$CSF = \sum_{i=1}^{N_c} exp(-d(c_i, c_i^*))$$

Here, c_i^* is nearest prototype of c_i and N_c is number of classes sampled in the task . We have used exponential of negative of distances, because $d(c_i, c_i^*)$ has large magnitude and normal loss function values are not comparable to it. As, we have to add these terms so we make them of similar order.

5.2.3 Modified Loss Function $J^*(w)$

We define new loss function as:

$$J^*(\phi) = -log(p_\phi(y = k|x)) + \lambda \times CSF$$

$$J^{new}(\phi) = -log(p_\phi(y = k|x)) + \alpha \times D$$
$$k(x, x') = -exp(-\frac{\|x - x'\|^2}{2\sigma^2})$$

Here, first term minimize the distance of data points from the cluster prototype within the cluster, whereas second term maximises the inter-cluster distances. λ is a hyperparameter.

5.3 Results

Type of Exp.	$\lambda = 5$	$\lambda = 10$	$\lambda = 15$	$\lambda = 20$
2 way 3 shot	76.84	80.75	79.76	79
2 way 5 shot	79.69	80.91	81.11	81.08
5 way 3 shot	53.91	52.56	53.71	53.54
5 way 5 shot	56.61	55.89	54.87	55.28

Table 4: Accuracies when using Cluster Separation Factor in the Loss Function, and using Euclidean distance metric

5.4 Inferences

From the above experiments we observe that adding the lambda parameter increases the accuracy of the model by upto 7%. We infer this happens because now the clusters are not close to each other, therefore the query points are now able to converge to nearest prototype with more confidence.

6 Conclusion and Future Works

We ran our experiment on several different approaches and we have concluded the following.

1. Among several distance metrics, L2 norm gave best results.
2. Using Medoid method to find the cluster prototypes decrease the model performance as medoid method requires large data to correctly identify the cluster prototypes which is not possible in Few-shot learning.
3. Using Cluster Separator Factor in the loss function increases the accuracy substantially. With Proper hyperparameter tuning of the the λ parameter, the accuracy can be further improved.

A proper hyperparameter tuning of the model would further increase the accuracy of the models. We believe there are many other possibilities left to explore in the domain of Prototypical Network.

The curse of dimensionality[9] is a problem which has always been associated with Prototypical Networks. In our future works we plan to address this issue. By addressing the curse of dimensionality and solving it, we hypothesize that the accuracy of the model can be further increased.

7 References

1. Prototypical Networks for Few-shot Learning, (19 June, 2017)

2. Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In Advances in Neural Information Processing Systems, pages 3630–3638, 2016.

3. Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. International Conference on Learning Representations, 2017.

4. Damien Francois, Vincent Wertz, and Michel Verleysen. 2007. The Concentration of Fractional Distances. IEEE Trans. on Knowl. and Data Eng. 19, 7 (July 2007), 873–886. https://doi.org/10.1109/TKDE.2007.1037

5. Mining of Massive Datasets,Ch-7, by Ullman.

6. Chen, Jiehua Hermelin, Danny Sorge, Manuel. (2018). On Computing Centroids According to the p-Norms of Hamming Distance Vectors.

7. Meta-Derm Diagnosis: Few-Shot Skin Disease Identification using Meta-Learning, Kushagra Mahajan, Monika Sharma, Lovekesh Vig.

8. 2012. Commun. ACM 55, 10 (October 2012).

9. Keogh E., Mueen A. (2017) Curse of Dimensionality. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning and Data Mining. Springer, Boston, MA. https://doi.org/10.1007/978-1-4899-7687-1$_1$92