Comparing Graph Architectures for Deep Metric Learning

Moritz Schüler
dept. of Informatics
Technical University Munich
Munich, Germany
moritz.schueler@tum.de

Luca Eyring
dept. of Informatics
Technical University Munich
Munich, Germany
luca.eyring@tum.de

Abstract

We compare different attention mechanisms leveraged in graph neural networks for the task of deep metric learning on the datasets CUB-200-2011 and Cars196. Herefore, we compare traditional multiplicative attention to the additive attention used in GAT and conduct an ablation study to verify the importance of different parts of the used architecture. We confirmed that the attention mechanism is indeed important for the task of deep metric learning. Additionally, we show how through the more expressive attention mechanism in GATv2 we can significantly increase performance on both datasets.

Keywords— deep metric learning, similarity learning, graph neural network, attention

1. Introduction

Deep metric learning tries to solve the task of attributing a similarity score between different data samples. In particular, it tries to find a lower dimensional mapping where the distance between data points depicts their similarity. Learning these mappings is challenging because taking into account the complete embedding space is often infeasible. Therefore, most methods only utilize the relation of two or three samples during training time by using appropriate losses together with techniques like hard negative mining.

However, work by Seidenschwarz et al. [8] provides ideas to tackle this issue and leverage the whole knowledge of the embedding space - at least within a mini-batch. A fully trainable setting is achieved by adopting a graph neural network together with a transformer architecture to the task of metric learning. In this work we build upon the latest research findings and compare different realizations of attention mechanism in graph neural network, discussing the similarities and differences. Additionally, we conduct an ablation study on the graph architecture to verify the importance of attention mechanism.

1.1. Related Work

The first works on deep metric learning compared embeddings with a distance function [3]. Improvements were made by the introduction of new loss functions like the Triplet loss [7], which unlike traditional losses compares not only two images, but provides a positive as well as a negative sample for each image. This helps to bring samples of the same class closer together while also pushing contradicting samples further apart in the embedding space.

Furthermore, the Group Loss [5] not only uses three data points, but the whole mini-batch to improve the ”clustering”. Whilst considering the global structure the refinement step of the Group loss is fixed and cannot be adapted according to the samples of the current mini-batch. To fight this issue [8] leveraged the power of graph neural networks. By constructing a graph and running several message passing steps the whole structure of a mini-batch can be exploited, whilst also allowing for fine-tuning of the refinement step. Another important part of the work was to use attention, as presented by [9], to automatically learn the relations between the samples within a mini-batch.

1.2. Research Objective

In this project, we want to build upon [8]. First, we want to compare its performance against graph attention models [2, 10]. We specifically investigate the different implementations of the attention mechanism, as work by [4] raised questions about the performance of attention without added linear layers. Therefore, we also conduct an ablation study to investigate the importance of the whole attention block, but also on the single attention layer, as well as the linear layers following the attention mechanism.

2. Methodology

In Figure 1 we show an overview of the used architecture introduced by [8] consisting of the following steps:

1. Feed each high dimensional image into a ResNet50 backbone and apply a linear layer to generate lower dimensional initial embedding feature vectors.
2. Construct a fully connected graph between all samples in a mini-batch and perform message-passing between nodes to refine initial embeddings by utlizing an attention mechanism followed by a MLP block.
3. Apply a final linear layer and perform classification to optimize both the MPN and the backbone CNN in an end-to-end fashion using a cross-entropy loss on the refined node feature vectors.

More formally, the backbone CNN followed by a linear layer is used to compute initial feature vectors $\mathbf{f} \in \mathbb{R}^d$ of dimension d. Afterwards, a fully connected graph is built using the whole mini-batch with initial node features $\mathbf{h}_i = \mathbf{f}$.

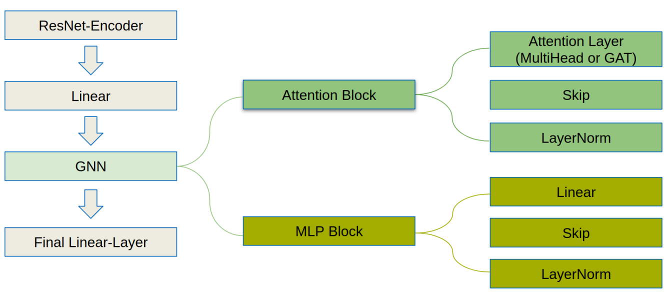

Figure 1. Overview of the used architecture

These feature embeddings are refined using multiple message passing steps where messages are passed between all neighbours $j \in N_i$ of node i, which in our case is all samples in the batch. Each of these steps consists of an attention block and an MLP block. In one message passing step we get without loss of generality the new representation $\mathbf{h}'_i$ of node i by:

$$\mathbf{h}'_i = \sum_{j \in N_i} \alpha_{ij} * \mathbf{W}_v h_j$$

Additionally, an attention mechanism represented by α_{ij} is added with the goal to compute a learned weighted average from neighbours $j \in N_i$ of a node i instead of weighting neighbours with equal importance. Here, we investigate different realizations of the attention mechanism. Over all of them these attention scores are normalized across all neighbours $j \in N_i$ using a softmax:

$$\alpha_{ij} = softmax_j(e(\mathbf{h}_i, \mathbf{h}_j)) = \frac{\exp(e(\mathbf{h}_i, \mathbf{h}_j))}{\sum_{j' \in N_i} \exp(e(\mathbf{h}_i, \mathbf{h}_j))}$$

where e is a scoring function that computes a score for every edge (j, i). This is where the implementations we are investigating differ. In [8] this mechanism is implemented through dot-product self-attention as introduced in Attention is all you need [9]:

$$\textit{Intra-Batch } [9]\text{: } e(\mathbf{h}_i, \mathbf{h}_j) = \frac{\mathbf{W}_q \mathbf{h}_i (\mathbf{W}_k \mathbf{h}_j)^T}{\sqrt{d}}$$

where d is the dimension of the embeddings and $\mathbf{W}_q$ is the weight matrix corresponding to receiving node i and $\mathbf{W}_k$ to the sending node. In all of the attention mechanisms a number of M attention heads are used and concatenated to form the updated feature embedding $\mathbf{h}'_i$. The dimensions of output and weight matrices of each attention head are scaled accordingly by $1/M$ to preserve the original embedding dimension.

As an alternative we evaluated Graph Attention Networks (*GAT*) [10], an architecture proposed solely to be applied on graphs with an additive attention mechanism:

$$\textit{GAT } [10]\text{: } e(\mathbf{h}_i, \mathbf{h}_j) = LeakyReLU(\mathbf{a}^T \cdot [\mathbf{W}\mathbf{h}_i || \mathbf{W}\mathbf{h}_j)]$$

where the weight vector $\mathbf{a} \in \mathbb{R}^{2d'}$ and $\mathbf{W} \in \mathbb{R}^{\mathbf{d'} \times \mathbf{d}}$ are learnable. Note that here only one weight matrix instead of two are used, and an activation function is added.

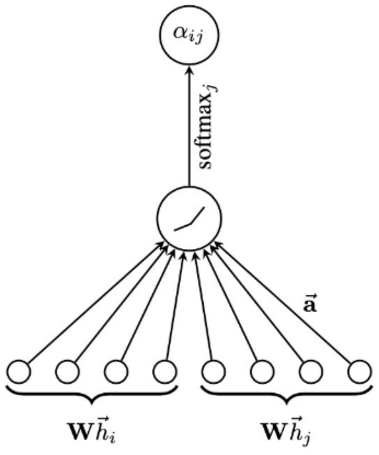

Figure 2. Additive attention mechanism used in *GAT*

Additionally, we also looked into the newly proposed Graph Attention Networks v2 *(GATv2)* [2]. The authors argue that the expressiveness of the attention functions is limited because the layers **W** and **a** are applied consecutively without an activation function in between and thus can be collapsed into a single layer. To fix this, they propose to move the layer **a** after the nonlinearity:

$$\textit{GATv2 } [2]\text{: } e(\mathbf{h}_i, \mathbf{h}_j) = \mathbf{a}^T \cdot LeakyReLU(\mathbf{W}[\mathbf{h}_i || \mathbf{h}_j)]$$

They also show that *GAT* tends to compute a global ranking of influential nodes while *GATv2* can learn different rankings of neighbours which is desired in our case.

In addition to the computation of the attention mechanism, a skip connection is added and a layer normalization is applied. The output is then fed into a MLP block consisting of two linear layers together with a skip connection and layer normalization. This then completes one message passing step and the output is fed into the next one.

Lastly, the output of the message passing steps gets processed by a final linear layer for classification to complete the fully optimizable pipeline.

3. Experiments

To get reliable and repeatable results we first rewrote part of the framework by [8] to achieve a fully deterministic implementation of the training procedure. Likewise, we adopted the existing implementations of *GAT* and *GATv2*.

We conducted an ablation study to evaluate the importance of the different parts of the architecture proposed in [8]. Here analyzed the following settings:

1. Only finetune the ResNet50 backbone.
2. Remove attention and MLP block and only train the backbone together with the final linear layer.

3. Ablate attention block.
4. Ablate MLP block.

In contrary to the results from [4], the attention mechanism proved important to the performance for deep metric learning. The quantitative results are presented in more detail in section 5.

Due to these positive results, we conducted further studies about different available realizations of the attention mechanisms. In particular we compare the original attention implementation from Vaswani et al. [9], used by Seidenschwarz et al. [8] with the specifically for graphs designed *GAT* and *GATv2*.

4. Evaluation

To asses the performance of the different architectures viable and commonly used evaluation criteria are needed. Therefore, we decided to benchmark all trained models on the, for retrieval tasks commonly used, metrics recall@k for several k, as well as normalized mutual information (NMI).

Recall@k is calculated by running a k-nearest neighbour algorithm and assigning a one to each sample, where the true class is within the k-nearest neighbours and a zero otherwise. To get the final score the zeros and ones are averaged. More loosely speaking, recall@1, for example, checks whether the nearest neighbour is from the same class as the sample under question.

Mutual Information evaluates the performance by running two kmeans clusterings and comparing their alignment. More precisely, one splits the data according to clusters, whereas the other splits according to class labels. This combats the creation of singletons, because the two splits would highly differ resulting in a low score. Furthermore, to get the NMI a bias-correction and normalization is applied to stay in the range of zero to one. Summarized, the NMI is a metric to attribute the goodness of a clustering.

In this work, our main focus and optimization criteria during training was recall@1, but the NMI scores are reported as a reference point as well.

5. Results

The results of the ablation study experiments over both datasets can be seen in tables 1 and 2. As a baseline we retrained the original architecture with our adjusted determinstic implementation. The achieved results performed worse compared to the ones reported in the paper [8]. However, this is to be expected since we employed a deterministic training setting.

	CARS196			
Method	**R@1**	Diff	**NMI**	Diff
Baseline	**87.1**	-	**71.7**	-
Backbone only	86.1	-1.0	67.7	-1.7
No MLP & Att	86.6	-0.5	69.3	-1.4
No Attention	86.8	-0.3	72.2	+0.5
No MLP	87.8	+0.7	72.0	+0.3

Table 1. Ablation Study CARS

	CUB-200-2011			
Method	**R@1**	Diff	**NMI**	Diff
Baseline	**69.4**	-	**72.4**	-
Backbone only	67.7	-1.7	69.5	-2.9
No MLP & Att	69.1	-0.3	70.9	-1.5
No Attention	69.3	-0.1	71.8	-0.6
No MLP	67.9	-1.5	71.2	-1.2

Table 2. Ablation Study CUB

Overall it can be said, that ablating any component of the architecture lowers the expressiveness of the model.

Regarding the claim of [4] that the linear layer is the most important part of the attention block, we see contradicting results between the two datasets. For CUB we indeed get significantly lower scores. However for CARS, we can even see a boost in performance. Considering the removal of the attention block, a small but consistent drop is present, indicating the importance of it. Interestingly, the biggest impact is found by ablating the message passing network, providing proof for the superior performance of the work by Seidenschwarz et al. [8] over Group Loss by having a trainable refinement step.

	CARS196			
Method	**R@1**	Diff	**R@1**	Diff
Baseline	**87.1**	-	**71.7**	-
Intra-Batch Paper	88.1	+1.0	74.8	+3.1
GAT	87.2	+0.1	**73.8**	+2.1
GATv2	**88.2**	+1.1	73.0	+1.3
GATv2 & no MLP	87.8	+0.7	71.7	+0.0

Table 3. Attention Mechanism CARS

	CUB-200-2011			
Method	**R@1**	Diff	**R@1**	Diff
Baseline	**69.4**	-	**72.1**	-
Intra-Batch Paper	70.3	+0.9	74.0	+1.6
GAT	68.9	-0.5	**72.8**	+0.7
GATv2	**69.8**	+0.4	72.4	+0.3
GATv2 & no MLP	69.5	+0.1	71.2	-0.9

Table 4. Attention Mechanism CUB

After the proof of the importance of the MPN as well as the attention, we conducted more experiments with different realizations of the attention mechanism, especially designed for graph networks. In table 3 and 4 we can see the respective results for our attention mechanism experiments on the CARS and CUB dataset. Training with the original *GAT* slightly boosts the recall@1 for the CARS problem, but lacks performance on CUB. The improved *GATv2* on the other hand outperforms the baseline by a significant margin on both datasets. For CARS it even performs better than the state of the art paper results. This confirms the claim that the local awareness of the improved version aids to better clustering performance as stated by the authors [2] through a more expressive attention mechanism.

6. Discussion

In the previous section, we showed that for the task of deep metric learning the attention mechanism is indeed a valuable feature. Furthermore, the comparison of the different attention implementation confirmed the claim that a local influence of neighbouring nodes boosts performance for a clustering task.

Interestingly, the best performance with the GAT implementations were achieved by concatenating the outputs of the network, whereas Vaswani's implementation achieved the highest recall when averaging the output.

On top of that, we want to mention that the hyperparameters used for all experiments were taken from [8]. This means they are specific for Vaswani's attention mechanism, therefore we think that the performance boost can be increased further by optimizing the parameters for *GATv2*.

Additionally, the performance of the architecture seems to be at least to some level task dependent as we got very different results between datasets for some experiments.

7. Future Work

In this work we tried to increase the understanding about the good performance of the architecture by [8], as well as improve on it. Despite the positive results many further investigations are possible.

An interesting idea could be to replace or comprehend the attention mechanism by introducing edge attributes. The weight of the edges could provide a similar function as the attention mechanism, whilst allowing for a more efficient backbone or graph neural net. [1] presented a general framework to divide the message passing step into note-to-edge and edge-to-node updates which could then be used to effectively train the network.

Another interesting continuation would be to investigate the transfer abilities of the trained networks from one dataset to another. To effectively analyze this we suggest the following experiments:

1. Baseline results: Train the model on an evaluation dataset.
2. How does training between datasets impact performance: Train the model on a different dataset and afterwards use the trained model with a new final linear layer to fine-tune on the evaluation dataset.
3. How does the model generalize to other datasets: Same approach as *2*. However, freeze all layer except the last linear layer while fine-tuning on the evaluation dataset.

Additionally, the sample efficiency could be explored. This could be done by iterative reducing of the amount of training samples and comparing the performance to earlier results.

Lastly, as mentioned by [6], it might be interesting to investigate the current training approach versus an approach where the data is split into train, validation and test. In this particular comparison the generalization ability / performance of both approaches would be highly interesting. This could also be further enhanced by checking the generalization under the use of different optimization metrics. Especially mean average precision at R seems promising as proposed in the same paper.

References

[1] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks, 2018. 4

[2] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks?, 2021. 1, 2, 4

[3] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In J. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6. Morgan-Kaufmann, 1994. 1

[4] Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth, 2021. 1, 3

[5] Ismail Elezi, Sebastiano Vascon, Alessandro Torcinovich, Marcello Pelillo, and Laura Leal-Taixe. The group loss for deep metric learning, 2020. 1

[6] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check, 2020. 4

[7] Matthew Schultz and Thorsten Joachims. Learning a distance metric from relative comparisons. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16. MIT Press, 2004. 1

[8] Jenny Seidenschwarz, Ismail Elezi, and Laura Leal-Taix. Learning intra-batch connections for deep metric learning, 2021. 1, 2, 3, 4

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. 1, 2, 3

[10] Petar Velikovi, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Li, and Yoshua Bengio. Graph attention networks, 2018. 1, 2