UNCERTAINTY IN IMAGE RETRIEVAL WITH LAPLACE APPROXIMATION

Hans Christian B. Mikkelsen, Martin Illum, Oliver S. Olsen

Technical University of Denmark
Department of applied mathematics
computer science

ABSTRACT

The ability to map uncertainties within image recognition models presents a greater challenge, especially the ability to quickly implement these models. In the following paper, we have investigated the utility of Laplace Approximation in deep learning, which is one of the more uncommon variations of Bayes or deep ensembles. Specifically we show that Laplace Approximation can be implemented for metric-learning-problems, to model uncertainties, with post-training implementation. It is shown that implementing Laplace Approximation can improve performance, especially in out-of-distribution cases. Further implementation of the method is possible, and should be investigated in the future.

***Index Terms*—** Metric Learning, Laplace Approximation, Bayesian, image recognition, Deep Learning

1. INTRODUCTION

Content-based image retrieval (CBIR) systems show great potential in challenges such as face recognition [1], medical image analysis [2] and landmark recognition [3]. These CBIR systems typically embed images into high-level features and retrieve with a nearest neighbor search. While this is efficient, the retrieval does not take uncertainty into account, which can be problematic in safety-critical applications. To model the uncertainty, the most common practice is to apply an isotropic distributions to their models latent representations [4, 5]. However, with the reason release of Laplace Approximation software [6] can we make much more flexible distributions for our models latent space and therefore hopefully model the uncertainty better. We propose that we can model model uncertainty with the Laplace Approximation and improve image retrieval on out-of-distribution samples.

1.1. Related Work

In recent years, image retrieval has gotten more attention due to its many applications [7]. Recently, neural networks have made impressive progress on learning representations for image retrieval [8, 9, 10], but common for them is the choice of loss function used for training; it should ideally be a loss that will encourage good ranking.

Uncertainty learning in Deep Neural Network (DNN) has gotten more attention [11, 5, 12] because of the improvement of model robustness and interpretability. There are two main types of uncertainty: model uncertainty and data uncertainty. Model uncertainty is the concept of uncertainty of the model parameters given the training data. This type of uncertainty can be minimised by obtaining additional data [11, 12]. Data uncertainty accounts for the uncertainty in output whose main source is the inherent noise in input data and therefore cannot be removed with more training data [13]. Bayesian modeling provides a principled and unified approach to equipping models with robust uncertainty estimates [14]. Even though, Bayesian neural networks (BNNs) have not been used much in practice. Common criticisms of BNNs are that they are difficult to implement, delicate to tune, expensive to train, and hard to scale to modern models and datasets. For instance, popular variational Bayesian methods [15, 16] needs considerable changes to the training process and model architecture. Another challenge is that their optimization process is slower and typically more unstable unless delicately tuned [17]. Other methods, such as deep ensembles [18] and Monte Carlo dropout (MC dropout) [14] promise to bring uncertainty quantification to standard neural networks in simple manners. However, these methods either require a considerable cost increase compared to a single network, have limited empirical performance or an unsatisfying Bayesian interpretation [6] and they do not apply to image retrieval, as models typically do not have proper likelihood functions [4].

2. METHOD

2.1. Metric Learning

The idea behind metric learning is to map data to an embedding space, where data points are close to each other if the data points are similar and far apart if dissimilar. Finding similarity between data points can in general be achieved with embedding and classification losses [19].

2.1.1. Embedding losses

Pair and triplet losses is the foundation of two fundamental losses for metric learning. One of the best known pair based losses is the contrastive loss [20], which tries to make the distance between similar pairs ($d_{similar}$) smaller than some threshold (m_{sim}), and the distance between dissimilar pairs ($d_{dissimilar}$) larger than some threshold (m_{dis}):

$$L_{constrastive} = [d_{similar} + m_{sim}]_+ + [m_{dim} - d_{dissimilar}]_+ \tag{1}$$

The theoretical down side of this method is that the same distance threshold is applied to all pairs, even though there may be a large variance in their similarities and dissimilarities [19].

The triplet loss theoretically addresses this issue. A triplet consists of three parts, an anchor and two samples, where one of the samples are similar to the anchor and the other is dissimilar. The triplet loss attempts to make the distances from the anchor to the similar sample (d_{as}) smaller than the distances from the anchor to the dissimilar sample (d_{ad}), by a predefined margin (m):

$$L_{triplet} = [d_{as} - d_{ad} + m]_+ \tag{2}$$

This theoretically places fewer restrictions on the embedding space, and allows the model to account for variance in interclass dissimilarities [19]. An issue with triplet loss and contrastive loss is that they require a time consuming and performance-sensitive pair/triplet mining procedure to train the model. The mining techniques we use will be presented in the next section.

2.1.2. Mining

The idea behind mining is to find the best pairs or triplets to train on. Two broad approaches can be used to mine the best pairs or triplets: offline and online. Offline mining is performed before batch construction, so that each batch is made the most informative sample [19]. On the other hand, online mining finds hard pairs and triplets within each randomly sampled batch. Newer mining techniques such as semihard postive mining have been used to make more flexible embeddings and generalize better to new unseen data [21].

2.2. Laplace Approximation

The Laplace distribution is a continuous probability distribution named after Pierre-Simon Laplace. It has been argued that the Laplace Approximation (LA) is a simple and cost-efficient, yet competitive approximation method for inference in Bayesian deep learning. LA works by approximating the posterior with Gaussian distribution at a local maximum, with covariance matrix representing the local curvature. The main advantages of LA are that the local curvature can be approximated from the Hessian matrix, as well as local maximums can be obtained from standard *maximum a posteriori* (MAP) training of multiple neural networks [6].

The laplace posterior approximation can be defined as, given an classification dataset $\mathcal{D} := \{(x_n \in^M, y_n \in^C)\}_{n=1}^N$ where the weights are $\theta \in^D$ where θ_{MAP} is a *maximum a-posterior (MAP)* estimate:

$$\begin{aligned} p(\theta|\mathcal{D}) &\approx \mathcal{N}(\theta; \theta_{MAP}, \Sigma) \\ \textit{where: } \Sigma &= (\nabla^2_\theta \mathcal{L}(\mathcal{D}; \theta|_{\theta_{\text{map}}})^{-1} \end{aligned} \tag{3}$$

2.3. Evaluation Metrics

In this paper we will use textbfMean Average Precision at K (mAP@K). It measure the precision of the *K* nearest neighbors [4]. This metric takes into consideration the ratio of similar and dissimilar among its neighbors. The library we will use for calculating this metric is Kevin Musgrave library [22].

3. NETWORK AND ARCHITECTURE

First we train a simple neural network that consist of five fully connected linear layers with a Relu activation function in between each layer and with a batch size of 8 for metric learning, the loss function applied is Contrastive Loss, and the miner we use is Triplet Margin Miner [22]. The final output of the network is of two dimensions. After the network is trained the Hessian matrix is approximated based on using the Jacobi matrix from the loss function and the full network. The hyper-parameters of our network is then tuned *post-hoc*. Finally predictions are made from each of the tuned networks resulting in a distribution of predictions.

4. EXPERIMENTS

4.1. Data

The data used for this project is the Fashion MNIST dataset, containing article images—consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 gray-scale image, associated with a label from 10 classes [23].

The network was trained with a batch size of 16 on an nvidia a100 GPU, with 40GB memory. The model was trained for a total of 15 epochs, as the loss function stagnated.

4.2. Results

From Table 1 can we see that our models mAP@K score are better with LA than without LA for K equal to 1, 3 and 5. The model without LA performs best at K equal to 7 and with LA performs the model best at K equal to 5. It can also been seen that the mAP@K increase the bigger K gets for both with and without LA.

	ML without LA	ML with LA
mAP@1	0.5337	**0.5683**
mAP@3	0.6832	**0.7020**
mAP@5	0.7002	**0.7103**
mAP@7	**0.7015**	0.6987

Table 1. Mean Average Precision at K 1, 3, 5 and 5 for metric learning and metric learning with LA

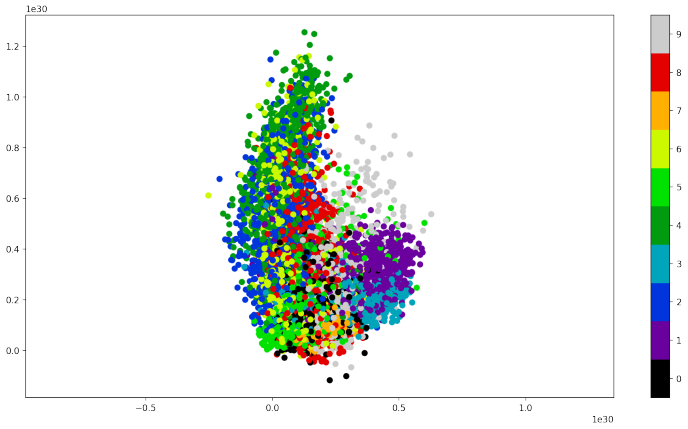

Fig. 1. Prediction feature visualization with la place approximation in two dimensions for the 10 different classes. From the label descriptions each color and the unique class it represents can be seen.

In Figure 1 and 2 all predictions are plotted in two dimensions with and without LA, where each color represents an unique class. A clear difference can be seen in the distribution of the two plots. The predictions with la place approximation are in general grouped closer together whereas for the prediction without la place, are more spread out and covers a significantly larger area. Both models from a graphic viewpoint seems to be able to distinguish some what between the classes, however the model with la place seems to be a little superior. The la place predictions seems to be vastly more certain at recognizing class 1, where as without the model without la place does better at class 0. It is clear that both models excels at certain classes, 1: trouser, 5: sandal, 9: boot, however the models finds it more difficult to distinguish between the classes 6: shirt, 2: Pullover, which are arguably clothing items which are more alike. In general both model seems to do well at distinguishing between clothing worn on different body parts, however shirts, t-shirts which more a like is a more difficult task. From the same figures, we also see that the LA model has tighter clusters for the out-of-distribution classes (7,8 9), where especially class 8 is significantly better.

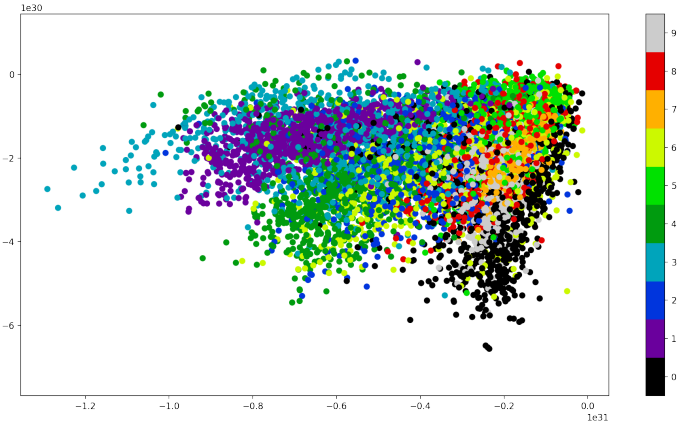

Fig. 2. Prediction feature visualization without LA in two dimensions for the 10 different classes. From the label descriptions each color and the unique class it represents can be seen.

The network has been trained with an in and out of distribution to be more robust which can be seen in Figure 4.2. It can be seen that the model has a lower variance on predictions for in distribution points, and higher variance for out of distribution.

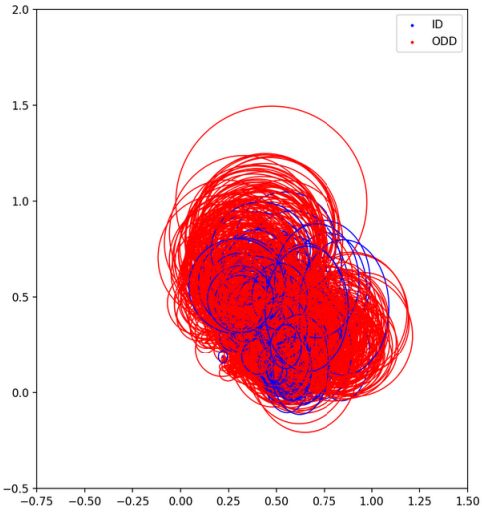

Fig. 3. Predictions of In Distribution (ID), and Out of Distribution (OOD), with the individual data-point variance, generated from the LA model. The model show 1500 randomly selected points from the test-set.

5. DISCUSSION

Overall, great results where not demonstrated with either traditional Metric Learning, nor with a LA implemented. Various reasons can have affected the results, and further research

is recommended.

It is generally accepted that convolution neural networks (CNN) are state of the art in terms of dealing with image data. Therefore it can be assumed that if one were to use CNN's instead of fully linear connected layers, better results in terms of accuracy could have been achieved. However, this was not possible due to the nature of CNN's and the method chosen to approximate the hessian matrix. A simple linear model was therefor chosen,to simplify the project, due to time-constraints, we where unable to test other model architectures, and compare performance. A CNN could have been used, where the LA was only applied on the linear layers, which might have yielded a higher accuracy.

We saw a small overall performance improvement was achieved with the application of LA, however whether the improvement can be justified by the small increase in accuracy can debated. We do however see an increase in performance of the out-of-distribution classes (7,8 9), using LA, which would indicate that the distribution has an effect on how the model places OOD classes.

It is however clear the computational cost using LA increases significantly, where around 95% of the running time is used on generation of the hessian matrix and making predictions on the newly generated networks. This could be reduced by only applying LA to the last layer of of the neural network or using the diagonal of the hessian instead of the full matrix.

In Table 1 we saw an increase in mAP@K, as k was increased, both with and without LA implemented. An explanation for such performance indicators, could be that our base model is not very good at separating the different classes in the latent space. This can also clearly be seen in Figure 1 and 2. This is another sign that a new base model architecture might be needed, since neither of the models have clear clusters for the 10 classes.

In Figure 4.2 it can be seen that the randomly sampled predictions have a fairly high variance, indicating that our model in general is quite uncertain which is once again discouraging. From the same plot we see that the variance of ID data is lower, with a higher variance for OOD data, which is to be expected.

Ideally we would have tested our general approach and model on multiple data sets, but was due to the limited time unable to do so. The same goes for our OOD tests, where we instead of looking at a completely different dataset, settled for classes in Fashion MNIST, and therefor limited our training set even further. It would also have been ideal to compare how the model compared with a LA model using online learning, which would allow us to evaluate if this implementations would have further benefit for out of distribution cases, or in distribution cases.

6. CONCLUSION

In this paper we have demonstrated that LA can be applied to metric learning. We have applied different metrics to evaluate the performance, but was unable to show a significant improvement with the implementation of LA, compared to traditional Metric Learning. We have approximated the Hessian matrix of our model, using the Jacobi matrix, and used said Hessian matrix, to produce 100 new networks from which we have made predictions. We have used traditional means of performance evaluations. It was seen that the only major difference between the a our model with and without la place was, that the model with la place performed better at predicting out of distribution cases. Concluded that a small increase in performance was achieved with our implementation. We also saw a benefit with implementation on OOD data. we conclude that further experiments needs to be conducted to conclude on the effectiveness of LA in metric learning.

7. FUTURE WORK

With regards to future work, the first step would be to experiment with our model architecture, because the initial structure is very simple, with only linear layers and Relu activation functions. It is general agreed upon that the CNN architecture is the best suited model architecture for image retrieval.

It would also be interesting to look further into if uncertainty modelling with LA would work on other dataset such as CUB 200-2011 [24] or Stanford Car-196 [25], which are in colors and have more classes than Fashion MNIST dataset.

Other evaluation metrics such as Recall at K (Recall@K) and Expected Calibration Error (ECE) would be beneficial to implement/examine to understand how LA models uncertainty even better, and to calibrate the model for better performance.

Further more we would like to compare the post-hoc approach with the more computational heavy online approach to see if we would get better results, and compare the benefits of the two approaches. As we already experienced a heavy computational cost with the post-hoc implementation of LA, we expect the online implementation to take up even more time and resources.

8. REFERENCES

[1] A. Martinez, "Face image retrieval using hmms," in *Proceedings IEEE Workshop on Content-Based Access of Image and Video Libraries (CBAIVL'99)*, 1999, pp. 35–39.

[2] Jayashree Kalpathy-Cramer, Alba García Seco de Herrera, Dina Demner-Fushman, Sameer Antani, Steven Bedrick, and Henning Müller, "Evaluating performance of biomedical image retrieval systems—an overview of the medical image retrieval task at imageclef 2004–2013," *Computerized Medical Imaging and Graphics*, vol. 39, pp. 55–61, 2015.

[3] Kohei Ozaki and Shuhei Yokoo, "Large-scale landmark retrieval/recognition under a noisy and diverse dataset," *CoRR*, vol. abs/1906.04087, 2019.

[4] Frederik Warburg, Martin Jørgensen, Javier Civera, and Søren Hauberg, "Bayesian triplet loss: Uncertainty quantification in image retrieval," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 12138–12148.

[5] Jie Chang, Zhonghao Lan, Changmao Cheng, and Yichen Wei, "Data uncertainty learning in face recognition," *CoRR*, vol. abs/2003.11339, 2020.

[6] Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig, "Laplace redux–effortless Bayesian deep learning," in *NeurIPS*, 2021.

[7] Liang Zheng, Yi Yang, and Qi Tian, "Sift meets cnn: A decade survey of instance retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 5, pp. 1224–1244, 2018.

[8] Relja Arandjelovic, Petr Gronát, Akihiko Torii, Tomás Pajdla, and Josef Sivic, "Netvlad: CNN architecture for weakly supervised place recognition," *CoRR*, vol. abs/1511.07247, 2015.

[9] Artem Babenko, Anton Slesarev, Alexander Chigorin, and Victor S. Lempitsky, "Neural codes for image retrieval," *CoRR*, vol. abs/1404.1777, 2014.

[10] Albert Gordo, Jon Almazán, Jérôme Revaud, and Diane Larlus, "Deep image retrieval: Learning global representations for image search," *CoRR*, vol. abs/1604.01325, 2016.

[11] Alex Kendall, Vijay Badrinarayanan, and Roberto Cipolla, "Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding," *CoRR*, vol. abs/1511.02680, 2015.

[12] Yarin Gal and Zoubin Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," 2015.

[13] Alex Kendall and Yarin Gal, "What uncertainties do we need in bayesian deep learning for computer vision?," *CoRR*, vol. abs/1703.04977, 2017.

[14] Yarin Gal and Zoubin Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of The 33rd International Conference on Machine Learning*, Maria Florina Balcan and Kilian Q. Weinberger, Eds., New York, New York, USA, 20–22 Jun 2016, vol. 48 of *Proceedings of Machine Learning Research*, pp. 1050–1059, PMLR.

[15] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra, "Weight uncertainty in neural networks," 2015.

[16] Alex Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, Eds. 2011, vol. 24, Curran Associates, Inc.

[17] Kazuki Osawa, Siddharth Swaroop, Anirudh Jain, Runa Eschenhagen, Richard E. Turner, Rio Yokota, and Mohammad Emtiyaz Khan, "Practical deep learning with bayesian principles," 2019.

[18] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. 2017, vol. 30, Curran Associates, Inc.

[19] Kevin Musgrave, Serge J. Belongie, and Ser-Nam Lim, "A metric learning reality check," *CoRR*, vol. abs/2003.08505, 2020.

[20] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, 2006, vol. 2, pp. 1735–1742.

[21] Hong Xuan, Abby Stylianou, and Robert Pless, "Improved embeddings with easy positive triplet mining," *CoRR*, vol. abs/1904.04370, 2019.

[22] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim, "Pytorch metric learning," 2020.

[23] Han Xiao, Kashif Rasul, and Roland Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," 2017.

[24] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie, "The caltech-ucsd birds-200-2011 dataset," 2011.

[25] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei, "3d object representations for fine-grained categorization," in *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.