Metric learning for fine grained classification

BAROUD Chikh Abdelghani
Paris Saclay university

***Abstract*—The application of fine-grained image classification can be problematic due to subtle differences between classes. The existing global feature-based methods have worse accuracies than regional feature-based methods, because regional feature-based methods focus on the determination of differentiated features within local regions. To learn more discriminative global features.in this project I proposed the use of metric learning with hard negative triplet which proved to be accurate to classify 6 mechanical pieces failures with an accuracy of 94% on testing data.**

INTRODUCTION :

Fine-grained classification is a challenging task in the field of computer vision, where the goal is to distinguish between visually similar categories within a broader class. This task is critical in various applications such as species identification, vehicle model recognition, and product categorization. Traditional classification methods often struggle to achieve high accuracy in fine-grained tasks due to the subtle differences between classes.

Metric learning has emerged as a powerful approach to address the challenges of fine-grained classification. By learning an embedding space where similar instances are closer together and dissimilar instances are farther apart, metric learning facilitates more accurate and robust classification. One of the most effective strategies within metric learning is the use of triplet loss, which trains the model to minimize the distance between an anchor and a positive example (from the same class) while maximizing the distance between the anchor and a negative example (from a different class).

In this project, we focus on enhancing metric learning for fine-grained classification through the use of hard negative triplet loss. Hard negatives are the most challenging examples that are currently being misclassified by the model, and incorporating them into the training process can significantly improve the model's discriminative ability. By emphasizing hard negatives, the model learns more robust features, leading to better performance in distinguishing subtle differences between fine-grained categories.

THIS REPORT DETAILS MY APPROACH TO INTEGRATING HARD NEGATIVE TRIPLET LOSS IN METRIC LEARNING FOR FINE-GRAINED CLASSIFICATION. WE DISCUSS THE THEORETICAL BACKGROUND, IMPLEMENTATION DETAILS, AND EXPERIMENTAL RESULTS THAT DEMONSTRATE THE EFFECTIVENESS OF OUR METHOD. THROUGH RIGOROUS EVALUATION, WE SHOW THAT MY APPROACH NOT ONLY enhances classification accuracy but also improves the model's generalization capabilities across various fine-grained datasets.

I. TOOLS USED IN THIS PROJECT

Google Colab: A cloud-based platform that provides a free environment for writing and executing code in Python, used because it provides easy access to powerful hardware resources, such as GPUs.
PyTorch: An open-source machine learning library for Python developed by Facebook's AI Research team, used for building and training deep neural networks.
Dataset: The dataset is failed mechanicel peices divided into two files: train and test, train has 1890 images and test has 389 images of size 256*256.
Google Drive: A cloud-based file storage and synchronization service provided by Google, used to store data.
Python: A popular high-level programming language widely used in various fields, including artificial intelligence.

II. CNN ARCHITECTURE

Resnet 50: In this project, we leverage the power of ResNet-50, a deep convolutional neural network renowned for its strong performance in image classification tasks, and integrate it with metric learning using hard negative triplet loss to enhance fine-grained classification. ResNet-50's architecture, featuring residual learning, allows for training very deep networks effectively, making it an excellent backbone for our approach.
Hard negatives are the most challenging examples that the model struggles to classify correctly and incorporating them into the training process can significantly improve the model's discriminative power. By focusing on hard negatives, the model learns more robust features, leading to

improved performance in distinguishing subtle differences between fine-grained categories.

This architecture aims to capture hierarchical representations of input images through convolutional layers and outputs an embeddings of size that i choose.

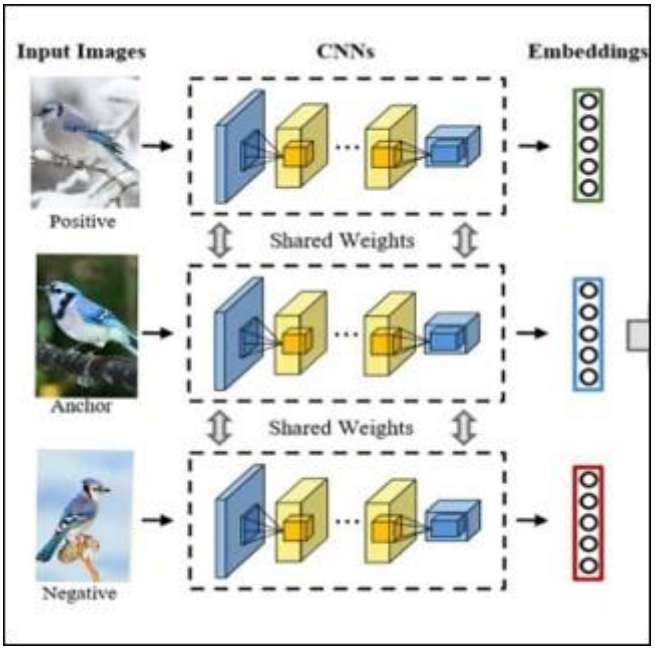

Fig 1:Resnet 50 Architecture for Image fine grained classification

The use of LeakyReLU activation functions helps alleviate the vanishing gradient problem, and batch normalization aids in stabilizing and accelerating the training process. Dropout is applied to prevent overfitting by randomly dropping out a fraction of neurons during training.

ReLU: Rectified Linear Unit, an activation function commonly used in neural networks, defined as: f(x) = max(0, x).

The main reason for using a convolutional neural network is to perform feature extraction from the images and reduce the size of the feature map at the end to fit it through a CNN network to apply a classification task.

Triplet loss: Triplet loss is a key component in metric learning, designed to improve a model's ability to distinguish between similar and dissimilar examples. It operates by using triplets of samples: an anchor, a positive example from the same class, and a negative example from a different class. The objective is to minimize the distance between the anchor and the positive example while maximizing the distance between the anchor and the negative example. This creates an embedding space where instances of the same class are clustered together, and those of different classes are spread apart, enhancing the model's discriminative power. Triplet loss is particularly effective in fine-grained classification tasks, where subtle differences between classes are crucial.

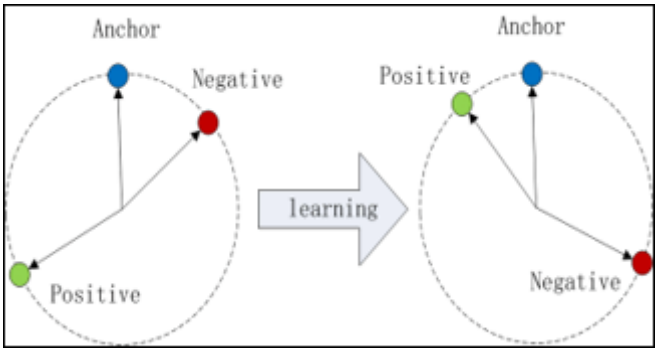

Fig 2: Triplet loss optimization process

Hard negative triplets:
In metric learning, the triplet loss function aims to ensure that an anchor is closer to a positive example (from the same class) than to a negative example (from a different class) by a certain margin. This relationship can be expressed through the following inequality:
$d(A,P)+\alpha<d(A,N)d(A,P)+\alpha<d(A,N)$

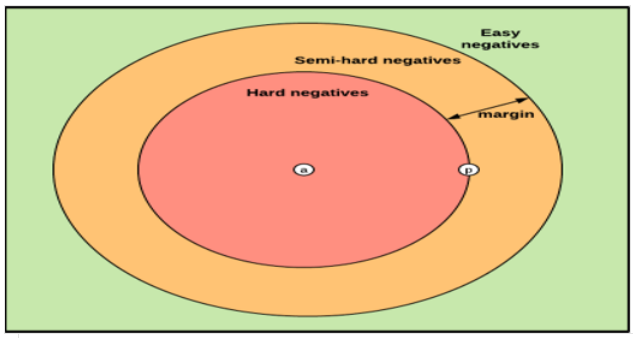

Fig 3:The three types of negatives, given an anchor and a positive

III. Training the Model:

To train a CNN model we must choose the loss function and the optimizer and define some hyper parameters to perform the training properly.

For this project we chose Stochastic gradient descent as an optimizer which is a popular optimization technique used in machine learning for finding the optimal parameters (weights and biases) in small batches, based on the loss calculated on each batch. To calculate the loss at each iteration, we used triplet loss. It operates by using triplets of samples: an anchor, a positive example from the same class, and a negative example from a different class. The objective is to minimize the distance between the anchor and the positive example while maximizing the distance between the anchor and the negative example.

$$Loss = \sum_{i=1}^{N} \left[\|f_i^a - f_i^p\|_2^2 - \|f_i^a - f_i^n\|_2^2 + \alpha \right]_+$$

Finally , we define some hyperparameters for our model which will effect its performance during training and can have a significant impact on the final results. Some hyperparameters include the learning rate to be 0.0001 Stochastic gradient descent Algorithm weight decay value is 0.05, the batch size equal to 16, and the number of epochs set to 10. After defining all these parameters, we start our training process on the train images and then we verify the accuracy of our model using the test images.

A. The result :

After testing the performance of our model, we end up with results mentioned in the table below:

Accuracy on the train images	Accuracy on the test images
98%	94%

The loss of train and test dataset:

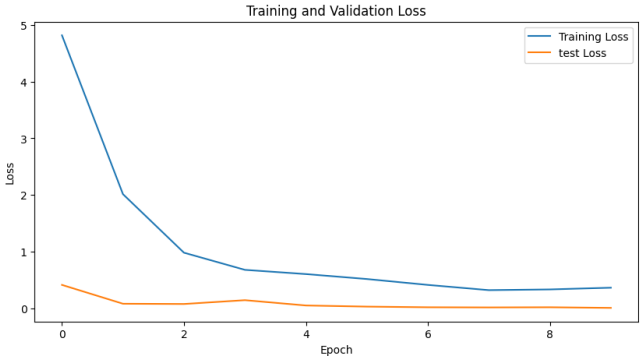

Fig 4: Training and testing losses

The accuracy of test dataset:

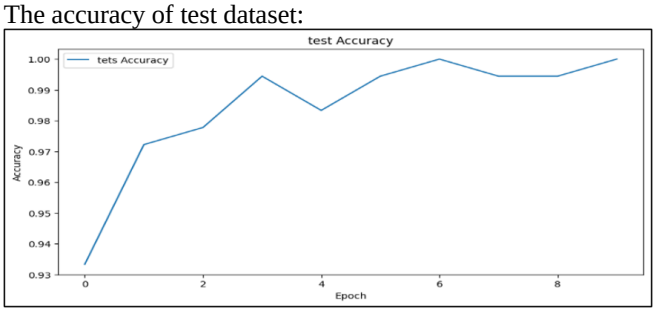

Fig 5:Test accuracy

As we can see the training data before and after training our model :

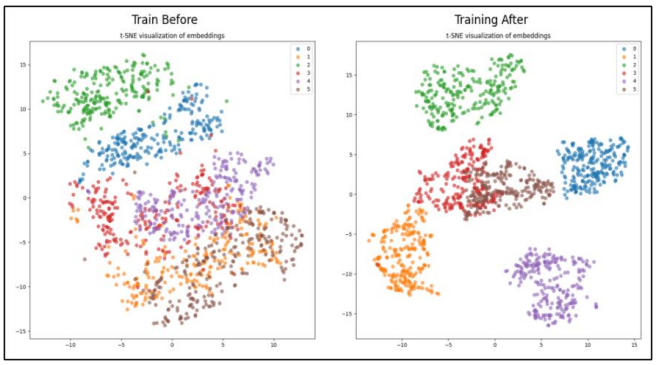

Fig 6 :Training embeddings before and after training plotted using TSNE

By choosing a margin of 0.2 and with hard positive negative triplet loss we can see that the 6 classes are well classified at the end.

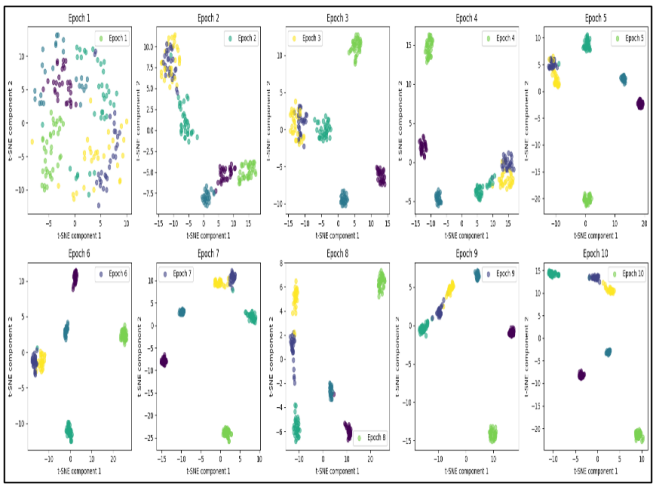

Fig 7:The evolution of the test embeddings assembling after each epoch

Mean cosine similarity:

For bath all triplet loss:

	Anchors and Positives	Anchors and Negatives
Mean Cosine Similarity	0.909957	0.085695

For hard negative triplet loss:

	Anchors and Positives	Anchors and Negatives
Mean Cosine Similarity	0.929957	0.065695

Fig 8 Mean cosine similariy

Discussion:

In this project, we successfully improved fine-grained classification by leveraging triplet loss with hard negative mining. Our approach yielded significant enhancements in the model's performance, achieving an impressive accuracy of 94% on the testing dataset and a mean cosine similarity of 0.92.

The use of triplet loss fundamentally altered how the model learned to distinguish between classes. By ensuring that the distance between an anchor and a positive example was minimized, while the distance between the anchor and a negative example was maximized, the model was able to develop a more discriminative embedding space. This was crucial for fine-grained classification, where the differences between classes can be very subtle.

The incorporation of hard negative mining further amplified the effectiveness of triplet loss. By selecting the most challenging negative examples for training, we forced the model to learn from the most difficult cases it was likely to encounter. This approach pushed the model to refine its feature representations, leading to a more robust and accurate classification performance.

The results demonstrate the efficacy of our method. Achieving a 94% accuracy on the testing data is a significant milestone, indicating that the model can reliably distinguish between fine-grained categories. Additionally, the mean cosine similarity of 0.92 reflects a high degree of consistency in the embedding space, confirming that the model places similar instances close together and dissimilar instances far apart.

These outcomes underscore the potential of combining triplet loss with hard negative mining in fine-grained classification tasks. By addressing the inherent challenges of such tasks with this advanced metric learning approach, we have set a new benchmark for accuracy and robustness. Future work can build on these findings by exploring different strategies for negative selection and further refining the model architecture to push the boundaries of fine-grained classification even further.

Conclusion:

The integration of triplet loss allowed us to create a more discriminative embedding space, essential for fine-grained classification where subtle differences matter. The addition of hard negative mining pushed the model to learn from the most difficult examples, leading to more robust feature representations and improved classification performance. These results highlight the potential of advanced metric learning techniques in addressing the complexities of fine-grained classification tasks. Our method has set a new standard for accuracy and consistency, demonstrating that triplet loss with hard negative mining is a powerful tool for enhancing model performance in this domain.

Future research can build on these findings by exploring additional methods for negative selection and further refining model architectures. By continuing to innovate in this space, we can further push the boundaries of what is possible in fine-grained classification, unlocking new applications and improving existing ones.

Acknowledgments:

We would like to express our sincere gratitude to our professor Hedi Tabia for his invaluable guidance and expertise throughout this project. His insightful guidance and explanation have greatly contributed to the success of our work. We are truly grateful for his time and effort in monitoring us.