A Feature Generator for Few-Shot Learning

Heethanjan Kanagalingam[1], Thenukan Pathmanathan[1], Navaneethan Ketheeswaran[1], Mokeeshan Vathanakumar[1], Mohamed Afham[2], and Ranga Rodrigo[1]

[1] Dept. of Electronic and Telecommunication Engineering, University of Moratuwa, Sri Lanka.
[2] Technical University of Darmstadt, Germany.
heethanjanheetha@gmail.com

Abstract. Few-shot learning (FSL) aims to enable models to recognize novel objects or classes with limited labeled data. Feature generators, which synthesize new data points to augment limited datasets, have emerged as a promising solution to this challenge. This paper investigates the effectiveness of feature generators in enhancing the embedding process for FSL tasks. To address the issue of inaccurate embeddings due to the scarcity of images per class, we introduce a feature generator that creates visual features from class-level textual descriptions. By training the generator with a combination of classifier loss, discriminator loss, and distance loss between the generated features and true class embeddings, we ensure the generation of accurate same-class features and enhance the overall feature representation. Our results show a significant improvement in accuracy over baseline methods, with our approach outperforming the baseline model by 10% in 1-shot and around 5% in 5-shot approaches. Additionally, both visual-only and visual + textual generators have also been tested in this paper. The code is publicly available at https://github.com/heethanjan/Feature-Generator-for-FSL.

Keywords: Few-shot learning (FSL) · Feature generator · Embedding process · Class-level semantic features

1 Introduction

FSL is a challenging task in machine learning, where the goal is to recognize and classify objects with limited labeled data. Unlike traditional deep learning models that require large amounts of labeled data to achieve high performance, FSL aims to perform well even when only a few examples per class are available for training. This task is crucial for applications where data collection is expensive, time-consuming, or impractical, such as medical image processing, rare species identification, and personalized user experiences.

To address the limitations of traditional models in FSL scenarios, researchers have explored various techniques, including meta-learning [5, 27], which aims to learn how to learn by leveraging knowledge from a wide range of tasks to adapt quickly to new ones; metric learning [31], which focuses on learning a

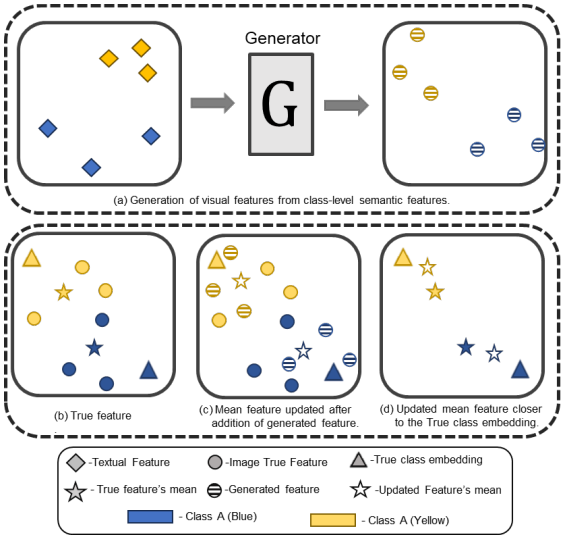

Fig. 1: The feature generation process. To generate the best optimum visual features from the class-level semantic features (a), the generated features are added to the initial true features(b), and the mean feature is updated by considering all the features (c). Finally, the updated features mean, obtained by combining the generated and real features, converges closer to the true class embedding (d).

similarity measure to effectively compare and distinguish between classes; and generative models [35, 42], which can synthesize new examples to augment the limited available data.

Even though there are various models to address the issues with FSL, there remains a significant gap in effectively integrating and leveraging the complementary information from textual and visual modalities. Current methods often treat these features independently, missing the opportunity to enhance the discriminative power of support class embeddings through their combined use. Additionally, existing generative models primarily focus on augmenting visual data without fully exploiting semantic information derived from class descriptions, which could provide valuable context and improve feature generation quality.

In this study, we propose a novel approach for FSL using a feature generator that leverages semantic features to generate visual features, thereby enhancing the support class embeddings. We contribute to synthesizing visual features from class-level textual descriptions using a novel feature generator. This approach combines semantic features with visual data and uses a combined loss function to align generated features with true class embeddings closely. By generating visual features from class-level semantic features, we aim to bridge the gap between textual and visual modalities, utilizing their complementary information to denote discriminative visual features better. This ensures that the classes in the support set are correctly represented in the embedding space. Fig. 1 shows our approach in detail.

Here, we selected textual descriptions that accurately represent the dominant visual features of each class. These descriptions highlight the critical features that distinguish one class from another. We ensured that consistent textual descriptions were used across both the training and test sets to maintain uniformity. Additionally, whether these descriptions were manually written or automatically generated, we followed a structured format to ensure coherence throughout the dataset.

The key idea behind our approach is to generate synthetic visual features, effectively transforming the n-shot learning scenario into a $2n$-shot learning scenario. This allows us to fine-tune the embedding of the support set classes and capture more discriminative and accurate information. Our approach utilizes a conditional generator, which takes the semantic feature of the support class as input and generates synthesized features. These generated features are then added to the support set, enhancing their representation.

To implement our approach, we employ a feature generator architecture comprising a classifier, discriminator, and generator. The classifier is trained to classify the true features, while the discriminator is trained to differentiate between true and generated features of the image. The generator, on the other hand, takes class-level semantic features as input and generates visual features that are closely aligned with the original features. This is achieved by training the generator to minimize a combined loss function that includes classifier loss, discriminator loss, and cosine distance loss during each feature generation step. Our ablation study (Section 4.3) shows that this combined loss leads to significantly higher accuracy than other loss combinations, confirming its effectiveness. By incorporating this cumulative loss approach, our method ensures that the generated features closely align with the original ones.

To evaluate the effectiveness of our approach, we conducted experiments on the miniImageNet [20] and tieredImageNet [21], which are commonly used benchmarks for FSL. We trained our generator with the Meta-Baseline [3] and Free Lunch [39] baselines, and accuracy significantly improved compared to those without the generator.

In summary, our proposed approach combines textual and visual information in the FSL setting by generating visual features from class-level semantic features.This approach not only leverages semantic features in a new way but also achieves a significant performance boost, particularly in 1-shot scenarios, which surpasses many existing state-of-the-art methods.Our experiments validate the effectiveness of our approach and demonstrate that it can be just used as a module in any baselines to improve their performance and accuracy in the few shot settings.

2 Related Work

FSL has gained significant attention as a promising approach for classification tasks when limited labeled training data is available [2, 9–12]. FSL methods aim to leverage semantic information and generate representative features to enhance

the performance of few-shot classifiers. FSL approaches can be categorized into optimization-based methods [5, 8, 14, 16] and data augmentation based methods [1, 24, 33, 38].

Early works in Few-Shot Learning (FSL) primarily centered around optimization-based methods, which aimed to adapt models to new tasks with just a few gradient updates. These approaches, often built within meta-learning frameworks, focused on learning a good initialization for model parameters [5, 20].

In contrast, data augmentation-based methods sought to enhance model performance in data-scarce scenarios by generating additional training samples. For instance, FeLMi [22] emphasizes hard mixup augmentation by interpolating between data points, while Label Hallucination [7] generates labels for unseen classes. Cap2Aug [23] uses captions to guide augmentation, and Global Local-Aware Augmentation [25] focuses on maintaining semantic orthogonality.

Semantic features play a crucial role in bridging the gap between limited visual data and the rich information embedded in textual descriptions. Various works have explored the integration of semantic information to enhance few-shot classifiers. For instance, [36] proposed an adaptive cross-modal FSL approach that effectively combines visual and semantic information for classification tasks. Similarly, [27] introduced prototype networks that utilize semantic features to generate representative prototypes for each class. Furthermore, [29] explored the learning of compositional representations for few-shot recognition, emphasizing the importance of semantic information in enhancing classification performance. However, these methods primarily enhance existing visual features using semantic information rather than generating new visual features directly from textual descriptions.

Considering the implementation of generative models, [42] presented a novel approach for generating representative samples for few-shot classification using a conditional generative adversarial network. Their method analyzes the integration of fake samples in the FSL problem, which fails to generate more effective features, that can be solved by the semantic feature usage in the visual feature generation.

In [37], proposed generating representative samples for few-shot classification by leveraging semantic features and the variation autoencoder (VAE) model, demonstrating significant improvements in classification performance. Even though their method shows significant performance, their training strategy depends on the representative sample, this can become less effective when there are few or no representative samples for classes.

Considering the limitation of the method, our approach involves developing a unique generative model that generates visual features from class-level textual descriptions by considering the true feature's mean. Additionally, our method integrates a combined loss function, which is used to closely align the generated feature with the true class-level embedding.

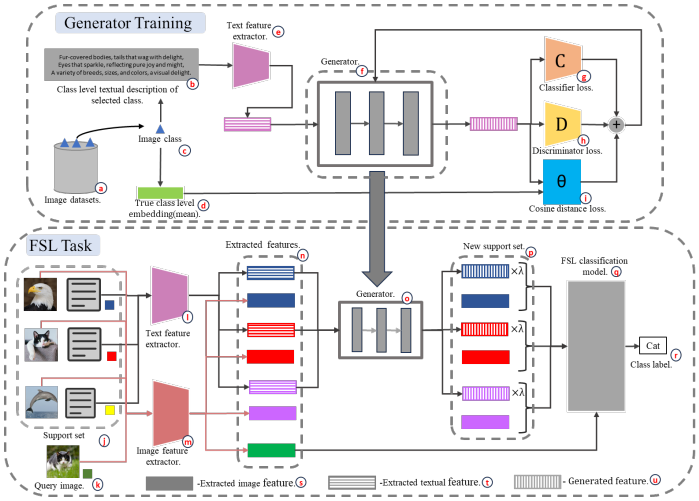

Fig. 2: The overall architecture diagram. To train the generator, an image true feature (c) and its corresponding class-level textual description (b) are taken from the image dataset (a). The true class embedding (d) is calculated by taking the mean of all the image true features belonging to the selected true feature (c) class. The generator (e) generates visual features (f) from the semantic feature (g) extracted from the class description (h) using a text feature extractor (i). The classifier loss (j)is calculated using categorical cross-entropy loss and discriminator loss (k) is calculated using binary cross-entropy loss. In contrast, the cosine distance loss (l) is computed as the distance between the true class embedding (m) and the generated feature. During training, the generator aims to minimize the sum of these three losses (j, k, l). The inference support set (n) contains Images and corresponding class descriptions. Semantic features (g) and visual features (o) are extracted using a text feature extractor (p) and an image feature extractor (q), respectively. The synthetic visual features (f) are generated by inputting the semantic features (g) to the generator (r). The generated feature is multiplied by λ and added to the new support set (s), which is subsequently used for the FSL classification task.

3 Approach

3.1 Notation

Suppose N labeled features belonging to n classes are provided for training, we can define the training dataset pairs (D_{tr}), semantic training pairs (S_{tr}) and the true class embedding training pairs (T_{tr}) are as follow:

$D_{tr} = \{(x_0, y_0), .., (x_i, y_i), ..., (x_N, y_N)\}$, where $x_i \in X$ denotes the feature and $y_i \in Y$ is the corresponding class label.

$S_{tr} = \{(s_0, l_0), ..., (s_k, l_k), ..., (s_n, l_n)\}$, where $s_k \in S$ denotes the class-level semantic feature and $l_k \in L$ denotes the corresponding class label.

$T_{tr} = \{(t_0, l_0), ..., (t_k, l_k), ...(t_n, l_n)\}$, where $t_k \in T$ denotes the true class embedding.

Considering the feature generator, $G(s_k) = \widetilde{x_k}$, where $\widetilde{x_k}$ is a generated feature from the generator.

3.2 Overall Pipeline (Textual Feature Generator)

Fig. 2 illustrates our generator training approach in detail. In our textual feature generator for FSL, we introduce the integration of semantic features extracted from class-level descriptions. This integration aims to leverage the complementary information from both modalities to enhance the embedding process, as image-true features and generated features are used to get the final support feature embedding. To achieve this task, We employ a feature generator architecture, that takes a semantic feature as input and generates visual features from it. Here, λ is used to balance the combination of the generator's output with the actual features in the support set.

Class-level descriptions: For each class, we construct a class-level description consisting of K sentences that describe the classes (some examples of the descriptions are provided in the supplementary document). We employ a text encoder to convert the class-level descriptions into semantic features. The resulting K semantic features from K descriptions are then averaged to obtain a single class-level semantic feature, passed through the generator network to generate a visual feature. To train the generator, we adopt a multi-loss training strategy that incorporates the objectives of the classifier, discriminator, and cosine distance between the generated feature and the true class embedding of a class. Here, the true class embedding is the mean of all the image true features belonging to a particular class in the training data set.

Classifier: The classifier is trained in parallel by inputting image true features and their corresponding labels using the categorical cross-entropy (CCE) loss between predicted class labels of the input image true features and the true class labels (Eq. (4)). During training, by minimizing the loss, the classifier improves its ability to classify input feature classes correctly. The (CCE) is :

$$CCE(P, T) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{C} T_{ij} \log(P_{ij}) \tag{1}$$

Where C is the number of classes, T_{ij} is the target for class j of sample i (one-hot encoded), P_{ij} is the predicted probability for class j of sample i.

Then the classifier loss L_c:

$$L_c = CCE(C(X), Y) \tag{2}$$

where $C(x)$ is output of the classifier for input feature x

Discriminator: The discriminator is trained to classify between the image's true features and generated features by inputting generated features and the image's true features in parallel. To achieve this, the discriminator loss comprises two key components: the real loss and the fake loss. The real loss evaluates the discriminator's ability to classify image true features correctly. It employs the binary corss entrophy(BCE) loss, measuring the disparity between the discriminator's prediction for input image true features and the target label, which is set to 1 to represent an image's true features. Conversely, the fake loss assesses the discriminator's aptitude in distinguishing generated samples as fake. It also employs BCE loss, comparing the discriminator's prediction for inputted generated features with the target label set to 0 to denote a fake sample. The overall discriminator loss is obtained by summing the real and fake losses together. By minimizing this loss during training, the discriminator enhances its ability to differentiate between real and fake samples, contributing to the overall effectiveness of the generative model.
The discriminator loss (L_d) is:

$$L_d = BCE(D(X), 1) + BCE(D(\widetilde{X}), 0) \tag{3}$$

Where $D(x)$ is output from the discriminator for input x. Here BCE is the binary cross entropy, calculated using the following equation.

$$BCE(P, T) = \frac{1}{N}\sum_{i=1}^{N} T_i log(P_i) + (1 - T_i) log(1 - P_i) \tag{4}$$

where P_i is prediction and T_i is true label

Cosine distance between the generated feature and the true class embedding: Here the distance is calculated to identify how far away the generated features are from the true class embedding.
cosine distance loss (CDL) :

$$CDL(A, B) = \frac{1}{N}\sum_{i=1}^{N}\left[1 - \frac{\mathbf{A}_i.\mathbf{B}_i}{\max(\|\mathbf{A}_i\|_2.\|\mathbf{B}_i\|_2, \epsilon)}\right] \tag{5}$$

Where A, B are points in embedding space.

Generator: The overall loss function of the generator combines three important components. Firstly, the classifier loss, which uses the categorical cross entropy(CCE) loss between the predicted class labels for the input generated features and the true class labels. By incorporating this loss, the generator learns to generate features that align with the correct class. Secondly, the discriminator loss uses BCE loss between the prediction for the generated feature and a target label that is always set to 1. It guides the generator to generate features that are indistinguishable from image true features (making generated features that

get classified as image true features by the discriminator). Lastly, the distance loss is computed as the cosine distance between the generated feature and the true class embedding. This loss motivates the generator to generate features that are closer to the true class embedding, allowing it to generate prominent visual features.

By incorporating semantic features and the classifier, discriminator, and embedding distance losses, our generator is trained to generate features that align with class labels and exhibit discriminability and closeness to the true embedding features. This approach enhances the embedding process in FSL, enabling improved generalization and recognition performance when presented with novel classes and limited labeled data.

Distance loss L_θ:

$$L_\theta = CDL(A, B) \tag{6}$$

$$L_{\theta g} = CDL(\widetilde{x}, \widetilde{T}) \tag{7}$$

$$L_{dg} = BCE(D(\widetilde{x}), 1) \tag{8}$$

$$L_{cg} = CCE(C(\widetilde{x}), y) \tag{9}$$

Here, $L_{\theta g}, L_{dg}$ and L_{cg} represent the distance loss, discriminator loss, and classifier loss of the generated features respectively.Then the total Generator loss L is given by:

$$L = L_{\theta g} + L_{dg} + L_{cg} \tag{10}$$

$$L = CDL(\widetilde{x}, \widetilde{T}) + BCE(D(\widetilde{x}), 1) + CCE(C(\widetilde{x}), y) \tag{11}$$

4 Experiments

4.1 Experimental Settings

Datasets: We evaluate our method using two commonly used benchmark datasets for FSL: miniImageNet and tieredImageNet. miniImageNet is a subset of the ILSVRC-12 dataset, which is widely used for image classification tasks. It consists of 100 different classes, with each class containing 600 images. The 100 classes are split into three sets: 64 base classes for pre-training, 16 validation classes for model evaluation during training, and 20 novel classes for final testing. tieredImageNet, on the other hand, is a larger subset of the ILSVRC-12 dataset. It contains 608 classes that are sampled from a hierarchical category structure. Each class in tieredImageNet has an average of 1281 images. The dataset is first divided into 34 super-categories, which are then further split into 20 classes for training, 6 classes for validation, and 8 classes for testing. This results in a total of 351 actual categories used for training, 97 categories for validation, and 160 categories for testing. By evaluating our method on these datasets, we can

assess its performance in FSL scenarios and compare it to other state-of-the-art approaches. The goal is to train models that can effectively classify images from novel classes with only a limited number of examples, mimicking the challenges of real-world FSL applications.

Implementation: Our generator is trained solely on the training sets of miniImageNet and tieredImageNet to avoid exposure to test set classes, ensuring consistent result comparison. The generator is initialized with weights from a visual-only generator, trained on 38,400 miniImageNet or 449,631 tieredImageNet image features, which improves its stability.

Our textual feature generator was trained separately with the train image features, which are extracted using respective feature extractors from respective baselines. For Meta-Baseline [3] and Free Lunch [39], the ResNet12 backbone is used. The dimension of the feature representation is 512 for both Meta-Baseline and Free Lunch. Here we used three sentences for each class, and we used the Clip [19] encoder to extract semantic features from those descriptions, and the textual feature dimension is 512. The architecture of the generator consists of three fully connected layers, each followed by a batch normalization layer and a LeakyReLU activation function.

The architecture of the discriminator consists of three fully connected layers, each followed by a batch normalization layer and a LeakyReLU activation function. The final layer projects the data from 128 dimensions to a single output, which represents the probability of the input being a real image. The sigmoid activation function is then used to squash the output into a range between 0 and 1, representing the probability of the input being real or fake.

The architecture of the classifier consists of two fully connected layers, each followed by a batch normalization layer and a LeakyReLU activation function. The final linear layer transforms the features from the 256-dimensional space to match the number of classes in the training set, which changes for each dataset, which is 64 for miniImageNet and 351 for tiredImageNet. This transformation enables the model to output class probabilities for each input sample. The final layer uses a sigmoid activation function, which squashes the values between 0 and 1. This activation function is applied to each class probability, representing the confidence or likelihood of the input sample belonging to each class.

Here Adam optimizer is used for the generator, discriminator, and classifier models, where the initial learning rate is 1e-4 for all three. All three models are trained using A6000 GPU for 5000 epochs at the same time as the generator is trained with the total generator loss, the discriminator is trained using the discriminator loss, and the classifier is trained using the classifier loss. It took around 10 hours to train all three models for the miniImageNet dataset.

Baseline methods: Our feature generator is a simple plugin module that can be trained externally and then can be used by connecting directly with the baseline architecture. Here we used 2 baseline architectures: Meta-Baseline and Free-Lunch. In this ResNet-12 is used as the backbone.

4.2 Results

Table 1: Comparison to prior works on miniImageNet and tieredImageNet.

Method	Backbone	miniImageNet		tieredImageNet	
		1-shot	5-shot	1-shot	5-shot
Matching Net [30]	ResNet-12	65.64 ± 0.20	78.72 ± 0.15	68.50 ± 0.92	80.60 ± 0.71
MAML [5]	ResNet-18	64.06 ± 0.18	80.58 ± 0.12	-	-
SimpleShot [32]	ResNet-18	62.85 ± 0.20	80.02 ± 0.14	69.09 ± 0.22	84.58 ± 0.16
CAN [6]	ResNet-12	63.85 ± 0.48	79.44 ± 0.34	69.89 ± 0.51	84.23 ± 0.37
S2M2 [17]	ResNet-18	64.06 ± 0.18	80.58 ± 0.12	-	-
TADAM [18]	ResNet-12	58.50 ± 0.30	76.70 ± 0.30	62.13 ± 0.31	81.92 ± 0.30
AM3 [36]	ResNet-12	65.30 ± 0.49	78.10 ± 0.36	69.08 ± 0.47	82.58 ± 0.31
DSN [26]	ResNet-12	62.64 ± 0.66	78.83 ± 0.45	66.22 ± 0.75	82.79 ± 0.48
Variational FSL [41]	ResNet-12	61.23 ± 0.26	77.69 ± 0.17	-	-
MetaOptNet [13]	ResNet-12	62.64 ± 0.61	78.63 ± 0.46	65.99 ± 0.72	81.56 ± 0.53
Robust20-distill [4]	ResNet-18	63.06 ± 0.61	80.63 ± 0.42	65.43 ± 0.21	70.44 ± 0.32
FEAT [40]	ResNet-12	66.78 ± 0.20	82.05 ± 0.14	70.80 ± 0.23	84.79 ± 0.16
RFS [28]	ResNet-12	62.02 ± 0.63	79.64 ± 0.44	69.74 ± 0.72	84.41 ± 0.55
Neg-Cosine [15]	ResNet-12	63.85 ± 0.81	81.57 ± 0.56	-	-
FRN [34]	ResNet-12	66.45 ± 0.19	82.83 ± 0.13	71.16 ± 0.22	86.01 ± 0.15
FeLMi [22]	ResNet-12	67.47 ± 0.78	86.08 ± 0.44	71.63 ± 0.89	87.07 ± 0.55
Label Hallucination [7]	ResNet-12	68.28 ± 0.77	86.54 ± 0.46	73.34 ± 1.25	87.68 ± 0.83
Global-and Local-Aware Augmentation [25]	ResNet-12	67.25 ± 0.36	82.80 ± 0.30	72.25 ± 0.40	86.37 ± 0.27
Meta-Baseline [3]	ResNet-12	63.17 ± 0.23	79.26 ± 0.17	68.62 ± 0.27	83.74 ± 0.18
Free Lunch [39]	ResNet-12	58.28	77.26	67.88	83.91
Meta-Baseline with Generator (ours)	ResNet-12	**74.62 ± 0.21**	**80.92 ± 0.16**	75.28 ± 0.25	**89.72 ± 0.18**
Free Lunch with Generator (ours)	ResNet-12	66.51	78.45	**76.59**	84.72

Following the standard setting, we conduct experiments on miniImageNet and tieredImageNet, the results and the comparison between past methods are shown in Tab. 1. As most of the methods use ResNet-12 as the backbone and some with ResNet-18 with the same input image size, we can make a fair comparison between the models.

We can see a significant accuracy improvement in the Meta baseline and Free Launch for both datasets by integrating our module. For the 1-shot we received 8.1% to 12% accuracy improvement and for the 5-shot we received 1.2% to 5% accuracy improvement.

Here we can clearly note that the 1-shot has a significant accuracy improvement compared to the 5-shot. This is because the impact of the generated feature is high when the number of features in a support set class is less. Furthermore, we can clearly denote that, except for the 5-shot approach with the miniImageNet dataset, our method surpasses all the state-of-the-art methods by a significant margin.

4.3 Ablation and Analysis

Visual Generator: Here, rather than giving the class-level textual feature as the input, we tested by giving a visual feature as the input. For this purpose, we trained the generator by adding the real extracted feature to a noise vector,

which has a mean of 0.1 and a variance of 0.28. In this experiment, only the input feature is changed and we tested it in the Meta-Baseline, and we got an accuracy improvement as shown in Tab. 2 and Tab. 3.

Table 2: Comparison to prior works on miniImageNet with the integration of visual generators. Average 5-way accuracy (%) with 95% confidence interval.

Model	1-shot	5-shot
Meta-Baseline [3]	63.17 ± 0.23	79.26 ± 0.17
Meta-Baseline + Visual Generator (ours)	63.64 ± 0.23	79.69 ± 0.16

Table 3: Comparison to prior works on tieredImageNet with the integration of visual generator. Average 5-way accuracy (%) with 95% confidence interval.

Model	1-shot	5-shot
Meta-Baseline [3]	68.62 ± 0.27	83.74 ± 0.18
Meta-Baseline + Visual Generator (ours)	69.04 ± 0.26	83.43 ± 0.18

Visual + Textual Generator: Here, rather than giving only the visual feature or the class-level textual feature, we give the features that are a combination of both visual and textual features. By analyzing the experimental results depicted in Fig. 3, it can be observed that by varying the weight parameter (α) between visual and textual features, we can manipulate the contribution of each feature type to the overall accuracy of the model. Here, α is used to set the weight of the textual features in the generator training in the visual + textual feature generator. When α increases, the accuracy increases: showing the effect of textual feature. For the textual feature generator, α is set to 1.

The plot illustrates a positive correlation between the weight of the textual feature and the resulting accuracy. As the weight of the textual feature increases, the model's accuracy also shows an upward trend. This suggests that the textual feature is crucial in determining the model's accuracy in this particular context.

This finding aligns with the underlying hypothesis that the textual feature contains significant information. As a result, assigning a higher weight to the textual feature enhances the strength of generated visual features.

Ablation of losses: We conducted the ablation for 1000 epochs and added the results in Tab. 4. The results show that all loss combinations underperform

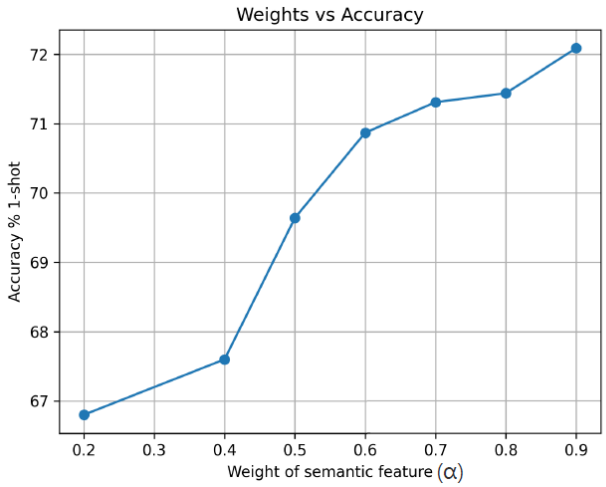

Fig. 3: Classification accuracy for different semantic weight α in miniImageNet dataset and Meta-Baseline as the baseline

compared to our combined loss in generator training, which achieved 68.59 ± 0.22 at 1000 epochs, confirming its effectiveness. The original training for 5000 epochs with the combined loss reached 74.62 accuracy.

Table 4: Ablation study on miniImageNet in a one-shot setting to confirm the need for our combined loss in the generator training

Losses	Classifier + Discriminator	Cosine + Discriminator	Classifier + Cosine
Accuracy	61.07 ± 0.22	64.19 ± 0.22	67.26 ± 0.21

4.4 Visualization of feature generator output

From the plots in Fig. 4, we can observe that the updated support class embeddings move closer to the true class embeddings. This shift occurs when a generated feature is added to the support set, validating the significant accuracy improvement observed in the baseline models. Additionally, we can clearly observe that the features generated using visual and textual features are more effective than the only visual features, which illustrates the need for textual features.

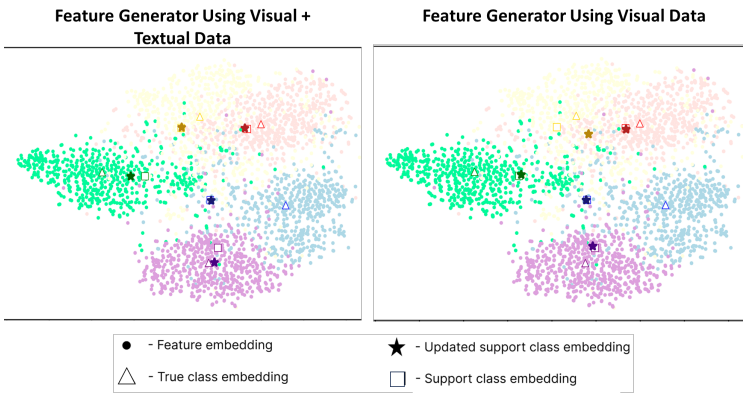

Fig. 4: Visualization of the effect of using textual features for visual feature generation. Here the updated support class embeddings move closer to the true class embedding.

5 Conclusion

In this paper, we presented an approach for FSL that integrates semantic features to enhance the embedding process for FSL tasks. We addressed the problem of inaccurate embeddings caused by few images per class by introducing a feature generator that generates visual features from textual class-level descriptions. Our approach utilized a combination of classifier loss, discriminator loss, and cosine distance loss to ensure the generation of accurate same-class features and improve the overall feature representation.

We demonstrated that the integration of semantic features significantly improves the alignment between generated and actual features, leading to better generalization and recognition performance. The integrated loss function with the generator ensures that the generated features are representative of the respective classes, thereby reducing the discrepancy between generated and true class embeddings.

Future work will focus on refining the feature generation process and exploring additional ways to incorporate semantic information. Additionally, we aim to test the scalability of our approach on more diverse and larger datasets to further confirm its robustness and general applicability.

Acknowledgement. The paper acknowledges the funds provided by the University of Moratuwa. The computational resources for this research were supported by the Accelerating Higher Education Expansion and Development (AHEAD) Operation of the Ministry of Higher Education, Sri Lanka, funded by the World Bank.

References

1. Antoniou, A., Storkey, A., Edwards, H.: Data augmentation generative adversarial networks. arXiv preprint arXiv:1711.04340 (2017)
2. Borowicz, A., Le, H., Humphries, G., Nehls, G., Höschle, C., Kosarev, V., Lynch, H.J.: Aerial-trained deep learning networks for surveying cetaceans from satellite imagery. PloS one **14**(10) (2019)
3. Chen, Y., Liu, Z., Xu, H., Darrell, T., Wang, X.: Meta-baseline: Exploring simple meta-learning for few-shot learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9062–9071 (2021)
4. Dvornik, N., Schmid, C., Mairal, J.: Diversity with cooperation: Ensemble methods for few-shot classification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3723–3731 (2019)
5. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International conference on machine learning. pp. 1126–1135. JMLR.org (2017)
6. Hou, R., Chang, H., Ma, B., Shan, S., Chen, X.: Cross attention network for few-shot classification. Advances in Neural Information Processing Systems **32** (2019)
7. Jian, Y., Torresani, L.: Label hallucination for few-shot classification (2021), `https://arxiv.org/abs/2112.03340`
8. Khadka, R., Jha, D., Hicks, S., Thambawita, V., Riegler, M.A., Ali, S., Halvorsen, P.: Meta-learning with implicit gradients in a few-shot setting for medical image segmentation. Computers in Biology and Medicine **143**, 105227 (2022)
9. Le, H., Nguyen, V., Yu, C.P., Samaras, D.: Geodesic distance histogram feature for video segmentation. In: Computer Vision–ACCV. pp. 275–290. Springer (2017)
10. Le, H., Samaras, D.: Shadow removal via shadow image decomposition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8578–8587 (2019)
11. Le, H., Samaras, D.: Physics-based shadow image decomposition for shadow removal. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(12), 9088–9101 (2021)
12. Le, H., Yu, C.P., Zelinsky, G., Samaras, D.: Co-localization with category-consistent features and geodesic distance propagation. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. pp. 1103–1112 (2017)
13. Lee, K., Maji, S., Ravichandran, A., Soatto, S.: Meta-learning with differentiable convex optimization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10657–10665 (2019)
14. Li, Z., Zhou, F., Chen, F., Li, H.: Meta-sgd: Learning to learn quickly for few-shot learning. arXiv preprint arXiv:1707.09835 (2017)
15. Liu, B., Cao, Y., Lin, Y., Li, Q., Zhang, Z., Long, M., Hu, H.: Negative margin matters: Understanding margin in few-shot classification. In: Computer Vision–ECCV. pp. 438–455. Springer (2020)
16. Liu, J., Song, L., Qin, Y.: Prototype rectification for few-shot learning. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16. pp. 741–756. Springer (2020)
17. Mangla, P., Kumari, N., Sinha, A., Singh, M., Krishnamurthy, B., Balasubramanian, V.N.: Charting the right manifold: Manifold mixup for few-shot learning. In: Proceedings of the IEEE/CVF Winter conference on applications of computer vision. pp. 2218–2227 (2020)

18. Oreshkin, B., Rodríguez López, P., Lacoste, A.: Tadam: Task dependent adaptive metric for improved few-shot learning. Advances in neural information processing systems **31** (2018)
19. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)
20. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning. In: International conference on learning representations (2017)
21. Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J.B., Larochelle, H., Zemel, R.S.: Meta-learning for semi-supervised few-shot classification. arXiv preprint arXiv:1803.00676 (2018)
22. Roy, A., Shah, A., Shah, K., Dhar, P., Cherian, A., Chellappa, R.: Felmi: Few shot learning with hard mixup. Advances in Neural Information Processing Systems **35**, 24474–24486 (2022)
23. Roy, A., Shah, A., Shah, K., Roy, A., Chellappa, R.: Cap2aug: Caption guided image to image data augmentation. arXiv preprint arXiv:2212.05404 (2022)
24. Schwartz, E., Karlinsky, L., Shtok, J., Harary, S., Marder, M., Feris, R., Kumar, A., Giryes, R., Bronstein, A.M.: Delta-encoder: an effective sample synthesis method for few-shot object recognition (2018), `https://arxiv.org/abs/1806.04734`
25. Shi, B., Li, W., Huo, J., Zhu, P., Wang, L., Gao, Y.: Global-and local-aware feature augmentation with semantic orthogonality for few-shot image classification. Pattern Recognition **142**, 109702 (2023)
26. Simon, C., Koniusz, P., Nock, R., Harandi, M.: Adaptive subspaces for few-shot learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4136–4145 (2020)
27. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. Advances in neural information processing systems **30** (2017)
28. Tian, Y., Wang, Y., Krishnan, D., Tenenbaum, J.B., Isola, P.: Rethinking few-shot image classification: a good embedding is all you need? In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16. pp. 266–282. Springer (2020)
29. Tokmakov, P., Wang, Y.X., Hebert, M.: Learning compositional representations for few-shot recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6372–6381 (2019)
30. Vinyals, O., Blundell, C., Lillicrap, T., kavukcuoglu, k., Wierstra, D.: Matching networks for one shot learning. In: Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 29. Curran Associates, Inc. (2016)
31. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. Advances in neural information processing systems **29** (2016)
32. Wang, Y., Chao, W., Weinberger, K., van der Maaten, L.S.: Revisiting nearest-neighbor classification for few-shot learning. arXiv preprint arXiv:1911.04623 (2019)
33. Wang, Y.X., Girshick, R., Hebert, M., Hariharan, B.: Low-shot learning from imaginary data. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7278–7286 (2018)
34. Wertheimer, D., Tang, L., Hariharan, B.: Few-shot classification with feature map reconstruction networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8012–8021 (2021)

35. Xian, Y., Lorenz, E., Schiele, B., Akata, Z.: Feature generating networks for zero-shot learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5542–5551 (2018)
36. Xing, C., Rostamzadeh, N., Oreshkin, B., O Pinheiro, P.O.: Adaptive cross-modal few-shot learning. Advances in neural information processing systems **32** (2019)
37. Xu, J., Le, H.: Generating representative samples for few-shot classification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9003–9013 (2022)
38. Xu, J., Le, H., Huang, M., Athar, S., Samaras, D.: Variational feature disentangling for fine-grained few-shot classification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8812–8821 (2021)
39. Yang, S., Liu, L., Xu, M.: Free lunch for few-shot learning: Distribution calibration. arXiv preprint arXiv:2101.06395 (2021)
40. Ye, H.J., Hu, H., Zhan, D.C., Sha, F.: Few-shot learning via embedding adaptation with set-to-set functions. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8808–8817 (2020)
41. Zhang, J., Zhao, C., Ni, B., Xu, M., Yang, X.: Variational few-shot learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1685–1694 (2019)
42. Zhang, R., Li, T., Li, X., Li, S., Goh, R.S.M., Ng, S.K.: Metagan: An adversarial approach to few-shot learning. In: Advances in neural information processing systems. pp. 2365–2374 (2018)