CVPR
#*****

Uncertainty Inclusive Contrastive Learning for Leveraging Synthetic Images

Anonymous CVPR submission

Paper ID *****

Abstract

Recent advancements in text-to-image generation models have sparked a growing interest in using synthesized training data to improve few-shot learning performance. However, prevailing approaches treat all generated data as uniformly important, neglecting the fact that the quality of generated images varies across different domains and datasets. This can hurt learning performance. In this work, we present Uncertain-inclusive Contrastive Learning (UniCon), a novel contrastive loss function that incorporates uncertainty weights for synthetic images during learning. Extending the framework of supervised contrastive learning, we add a learned hyperparameter that weights the synthetic input images per class, adjusting the influence of synthetic images during the training process. We evaluate the effectiveness of the UniCon-learned representations against traditional supervised contrastive learning, both with and without synthetic images. Across three different fine-grained classification datasets, we find that the learned representation space generated by the UniCon loss function incorporating synthetic data leads to significantly improved downstream classification performance in comparison to supervised contrastive learning baselines.

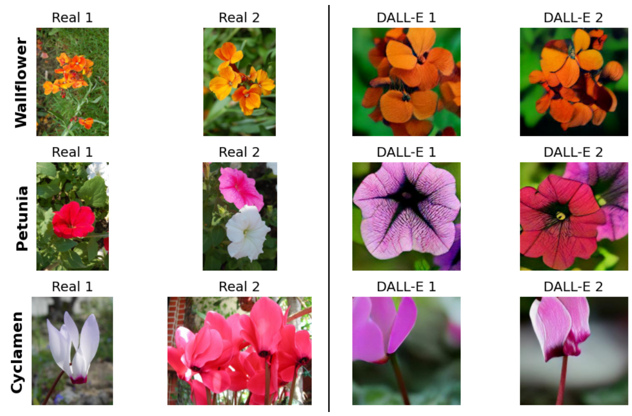


Figure 1. Examples of ground truth images from the Flowers10 dataset compared to the DALL-E generated images. The quality of DALL-E synthetic images varies by class. While synthetic wallflowers seem to correctly reflect the real data distribution, the synthetic petunias and synthetic cyclamens vary in color diversity and morphology from the real data, respectively.

1. Introduction

Powerful text-to-image generation models enable the synthesis of high-quality images from textual descriptions [4, 36]. These models have fueled research using synthetic data to provide additional support for various learning tasks [18, 24, 33, 41]. Training with synthetic images can help improve performance for challenging discriminative tasks relative to training on real images alone [1, 28]. Synthetic images are particularly beneficial when there is limited training data (i.e. few shot learning) since they can expand the training data set distribution and improve model performance on downstream tasks [9, 26].

While recent advancements have significantly improved synthetic image generation, the quality of generated synthetic images varies across diverse domains and datasets [11, 31, 32, 38]. Generative AI models often fail to capture pertinent attributes when generating images of fine-grained classes (e.g., flower species) [15]. Figure 1 showcases this phenomenon with examples of real images and DALL-E-generated images from corresponding classes in the Flowers102 dataset [17]. The variance in DALL-E's performance—high accuracy for wallflowers, misrepresented color diversity for petunias, and distorted morphology for cyclamen, highlights the challenges in generative AI when dealing with intricate patterns and complex colorations. This variability in capturing training data distribution can hinder accuracy in nuanced domains[12, 30]. Thus, understanding how and when to use synthetic support set samples in the learning process is crucial, especially for difficult vision tasks. Existing methods often treat synthetic images as if they were as informative as real images in the training process [1, 22]. Instead, we propose an approach that automatically adjusts the use of synthetic images based on their ability to improve performance.

We introduce Uncertainty-Inclusive Contrastive Learning

(UniCon), a novel contrastive learning framework designed to automatically learn uncertainty weights for synthetic images. Extending on supervised contrastive learning methods, we consider both positive and uncertain (synthetic) examples per anchor rather than only positive examples. The method down-weights uncertain examples if synthetic examples do not improve classification accuracy. Our key contributions in this paper include:

- We introduce UniCon, a contrastive learning method that automatically learning uncertainty weights for synthetic images.
- We show that UniCon improves few-shot classification performance in comparison to standard supervised contrastive learning (both using and not using synthetic images) on two different fine-grained datasets (Flowers10 and CUBS10) and two different methods of generating images (DALL-E and stable diffusion).
- Using synthetic data, we demonstrate that our learned weights are correlated with the relevance and quality of synthetic images in comparison to the real images.

2. Related Work

2.1. Text-to-Image Generation Models

Recent technological developments have led to the development of models capable of synthesizing highly realistic and contextually accurate images from textual descriptions [23, 40]. These developments include the introduction of autoregressive methods (e.g. DALL-E [18], PARTI [39]) that make use of large-scale image-text data during training. More recently, diffusion models have become the new state-of-the-art model for text-image synthesis [16, 23]. These diffusion models learn an estimation on Markov diffusion process using variational inference and are able to produce images with unprecedented detail, diversity, and fidelity to complex text prompts [19, 24, 25].

2.2. Generating Synthetic Data

Generative text-to-image models have increasingly been used to produce synthetic data for various machine learning tasks. Synthetic data can improve training performance on tasks from image classification and object detection by generating more diverse training datasets [9, 26]. This data augmentation is particularly crucial where real data is scarce or difficult to obtain. For example, GLIDE [16] generated images have been shown to improve performance, particularly in zero-shot and few-shot settings [9]. Other studies show that synthetic data augmentation strategies for medical images using GAN [7, 20] and diffusion models [2] can help improve medical image classification.

2.3. Weighting Synthetic Data

The idea of weighting training examples based on their informativeness or quality has been explored in various contexts, especially aimed at improving model performance and robustness. Meta-learning approaches have been developed to reweigh training examples based on their contribution to the model's performance, learning to assign higher weights to informative examples and lower weights to noisy or less relevant ones [13, 14, 21]. However, these methods do not address training with explicitly synthesized data. More recently, methods have been introduced to find optimal mixing ratios using more or less synthetic data for improving downstream performance [5, 37]. These methods still treat the resulting mixed training data equivalently. In the domain of leveraging synthetic data, Tsutsui et al. explored training an image fusion network mixing real and synthetic images using learned weights from a separate CNN network to create hybrid images [29]. This method, however, relies on training a separate network to fuse real and synthetic images, which can be computationally intensive and does not leverage synthetic images as-is. Although a weighting mechanism is used, the resulting hybrid images could potentially blur distinct features and reduce the learning benefits of using synthetic data in their unaltered state. Furthermore, the inherent characteristics of synthetic images could be more valuable for learning, either by offering distinctive features to contrast with real data distributions or serving as augmentative elements for existing real data.

While these methods breach the idea of taking note of the varying quality and diversity of synthetic data, an explicit and dynamic weighting approach may allow for more flexibility in adjusting the importance of synthetic images in how they inform the representations of their respective real classes.

3. Uncertainty-Inclusive Contrastive Learning

Problem Setup Consider an image dataset $X = \{x_1, x_2, ..., x_n\}$ with corresponding fine-grained k-class classification labels $\{y_1, ..., y_k\}$ where each $y_i \in \{C^1, ..., C^k\}$. For each class C^j, a set of synthetic images $U^j = \{u_1^j, u_2^j, ..., u_m^j\}$ is generated, and the union of these sets forms the synthetic image dataset $U = \bigcup_{j=1}^k U^j$. We aim to learn a classification f_θ that solves $y = f_\theta(x)$ for $x \in X$.

If we assume that U and X are generated from identical distributions, we can simply learn $y = f'_\theta(z)$ for $z \in U \cup X$ using standard supervised learning. Alternatively, if we assume the U is not useful for training, we can learn f_θ using only X, Y. However, in many real-world scenarios, the quality and relevance of the synthetic data U may vary, and it may not be optimal to either fully include or completely exclude U from the training process.

3.1. Supervised Contrastive Learning (SupCon) Loss

In the supervised contrastive learning setup, training proceeds by selecting a batch of N randomly sampled data $\{x_i, y_i\}_{i=1...N}$. We randomly sample two distinct label preserving augmentations, $\tilde{x}_{2i}$ and $\tilde{x}_{2i-1}$, for each x_i to construct $2N$ augmented samples, $\{\tilde{x}_j\}_{j=1...2N}$. Let $A(i) = \{1, ...2N\}\backslash i$ be the set of all samples and augmentations not including i. We define g to be a projection head that maps the embedding to the similarity space represented as the surface of the unit sphere $\mathbb{S}^e = \{v \in \mathbb{R}^e : ||v||_2 = 1\}$. Finally, we define $v_i = g(h_i)$ as the mapping of h_i to the projection space. Supervised contrastive learning encourages samples with the same label to have similar embeddings and samples with a different label to have different embeddings. We follow the literature in referring to samples with the same label as the anchor image x_i as the positive samples, and samples with a different label than that of x_i's as the negative samples.

After generating synthetic images, a natural question arises: how can we incorporate synthetic images in this supervised loss? Two trivial extensions include: 1) treating all synthetic images as real images (**SupCon-Mixed**) and 2) ignoring all of the synthetic images entirely (**SupCon-Real**). Note that these two extensions make sense if 1) synthetic images are not differentiable from the real images and 2) synthetic images are not useful for training, respectively. The loss function for SupCon-Real is:

$$\mathcal{L}_{SupCon} = \sum_{i \in I} \frac{-1}{|P(i)|} \sum_{p \in P(i)} log \frac{exp(\frac{v_i^T v_p}{\tau})}{\sum_{a \in A(i)} exp(\frac{v_i^T v_a}{\tau})} \tag{1}$$

where $|S|$ denotes the cardinality of the set S, $P(i)$ denotes the positive set with all other samples with the same label as x_i, i.e., $P(i) = \{j \in A(i) : y_j = y_i\}$, I denotes the set of all samples in a particular batch, and $\tau \in \{0, \infty\}$ represents a temperature hyperparameter.

We can extend Equation 1 to express the loss function for SupCon-Mixed as

$$\mathcal{L}_{SupCon^m} = \sum_{i \in I} \frac{-1}{|P_{U,C}(i)|} \sum_{p \in P_{U,C}(i)} log \frac{exp(\frac{v_i^T v_p}{\tau})}{\sum_{a \in A_{U,C}(i)} exp(\frac{v_i^T v_a}{\tau})} \tag{2}$$

where $P_{U,C}(i) := \{x_j : \forall x_j \text{ if } y_j = y_i\} \cup U_i$ and $A_{U,C}(i) = \{U \cup C - P_{U,C}(i)\}\backslash x_i$.

3.2. Uncertainty Inclusive Contrastive Learning (UniCon)

UniCon incorporates the intuition that it can be useful to weigh synthetic images less than real images, but not discount them entirely. The Uncertain Contrastive Learning (UniCon) loss modifies the SupCon-Mixed loss by adding a weighted term for all support set images U that correspond to an anchor input i. To achieve this, the UniCon loss includes class-specific weighting hyperparameters $\{w_i\}_{i=1}^{C}$, where C is the total number of classes in the dataset. Each class is allocated an individual uncertainty weight w_i, allowing for a tailored balance between real and synthetic data contributions for each class. The UniCon loss function is:

$$\mathcal{L}_{UniCon} = \sum_{i \in S(i)} (\frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(\frac{v_i^T v_p}{\tau})}{\sum_{a \in A(i)} \exp(\frac{v_i^T v_a}{\tau})} + \frac{-w_{y_i}}{|U(i)|} \sum_{u \in U(i)} \log \frac{\exp(\frac{v_i^T v_u}{\tau})}{\sum_{a \in A(i)} \exp(\frac{v_i^T v_a}{\tau})}) \tag{3}$$

Here, we consider the set S as all the real images in the batch such that $S \subseteq X$. Then for each anchor image $i \in S(i)$, $P(i)$ refers to the set of all indices of positive pairs from the same class that are real images so $P(i) \subseteq X$. The left term of the outer summation is identical to the SupCon-Real loss function.

The right term of the loss function introduces w_{y_i}, the weighting hyperparameter, where y_i corresponds to the class of the anchor image i. $U(i)$ refers to the set of all indices of inputs that are in the same class as anchor i but are synthetic support set images. Effectively, each normal input anchor i contributes to the UniCon loss through the sum of the SupCon loss and a weighted sum of all the similarities between the anchor and its corresponding support set images.

3.3. Bayesian Optimization for Optimal Hyperparameter Selection

We employed Bayesian optimization to learn the class-specific weighting hyperparameters $w_{i\,i=1}^{\,C}$ in the UniCon loss function, using the **gp_minimize** function from the Scikit-Optimize package [10]. The hyperparameters in this context refer to the class-specific weights w_i that control the influence of synthetic examples from each class in the UniCon loss function. Bayesian optimization is a method particularly suited for the optimization of complex, non-convex functions and consists of two primary components: a method for statistical inference, which is usually Gaussian process regression, and an acquisition function for deciding where to sample [27]. In our Bayesian Optimization process, we utilized a Gaussian Process (GP) as the surrogate model for its ability to handle the complexity and uncertainty of the objective function, efficiently capturing the nuanced relationship between hyperparameters and validation accuracy [6]. Bayesian Optimization progresses for n_{iter} calls to the objective function, and the GP model is updated with new data points obtained from objective function evaluations. For the acquisition function, we used 'gp_hedge', which probabilistically chooses among different strategies like Expected Improvement, Probability of Improvement, and Lower Confidence Bound in each iteration. The acquisition optimizer was set to 'lbfgs', a method known for its effectiveness in high-dimensional optimization problems [6]. All other parameters of the gp_minimize function were set to their default values.

The pseudocode for the objective function and Bayesian optimization process is as follows:

Algorithm 1 Bayesian Optimization for UniCon Hyperparameter Tuning

```
Initialize hyperparameter vector w = [w_1, ..., w_C]
Set number of contrastive runs n_contr
Set number of classifier runs n_classif
function OBJECTIVE FUNCTION F(w):
    AvgAcc ← 0
    for i = 1 to n_contr do
        Train Encoder_UniCon(w) to obtain E_i
        for j = 1 to n_classif do
            Acc_{i,j} ← Classifier_j(E_i)
        end for
    end for
    AvgAcc ← Average({Acc_{i,j}})
    return AvgAcc
end function
w* ← gp_minimize(F, space)
    Set iterations n_iter and initial points n_init
    Acquisition function: 'gp_hedge'
    Acquisition optimizer: 'lbfgs'
    Default values for remaining parameters
return w*                          ▷ Return optimal weights
```

The objective function $\mathcal{F}(\mathbf{w})$ evaluates the average validation accuracy across n_{contr} contrastive and n_{classif} classifier runs. For each contrastive run, the UniCon encoder is trained with the corresponding class-specific weights to obtain embeddings. Then, for each classifier run, a classifier is trained using the embeddings, and the validation accuracy is computed. The average accuracy across all runs is returned as the objective function value. The gp_minimize function iteratively evaluates the objective function at different hyperparameter configurations, updates the GP model, and searches for the optimal hyperparameter vector $\mathbf{w}^*$ that maximizes the average validation accuracy.

4. Experiments

In this section, we describe our experimental setup for evaluating the proposed UniCon method against supervised contrastive learning baselines on fine-grained image classification tasks. We compare the few-shot performance of UniCon to two baseline approaches, SupCon-Real and SupCon-Mixed. The experiments are conducted on subsets of two fine-grained datasets: Flowers102 and CUBS-200-2011 [17, 34]. Additionally, we performed additional studies on MNIST images where we used synthetic image classes of controlled quality to validate the effectiveness of the UniCon method and the expected learned weights [3].

4.1. Datasets

We evaluate our method on a subset of two classification datasets: Flower102 and CUBS-200-2011.

From the Flowers102 dataset, we used the ten largest classes to create a subset dataset called **Flowers10**: petunia, passion flower, wallflower, water lily, watercress, rose, frangipani, foxglove, cyclamen, and lotus. Each class had between 137 and 258 real images.

From the CUBS-200-2011 dataset, we used the ten largest classes to create a subset dataset called **CUBS10**: Laysan Albatross, Cardinal, Mangrove Cuckoo, Purple Finch, California Gull, Anna Hummingbird, Florida Jay, Baltimore Oriole, Brown Pelican, Common Raven. Each class had between 79 and 91 real images.

For the Flowers10 and CUBS10 dataset classes, we generated two sets of 32 synthetic images per class, using DALL-E and Stable Diffusion [15, 23]. Images were generated using a text prompt of "a photo of { }", where the blank was filled with the corresponding class names in plain text.

For a controlled study, we selected a subset of 400 images of the digit 0 and 400 images of the digit 9 from the MNIST dataset. For the support set images, we generated 300 images that were morphs of the digits 0 and 9. To generate morphed images that blend the characteristics of two distinct classes, we introduce a morphing equation controlled by the parameter ρ. Through this process, each morphed image is created by merging an image from class C_0 (images of 0s) and class C_9 (images of 9s), according to the equation:

$$M(\rho) = \rho \cdot I_{C_0} + (1 - \rho) \cdot I_{C_9} \tag{4}$$

In this equation, M is the morphed image, I_{C_0} and I_{C_9} are images from class C_0 and class C_9 respectively, and ρ is the morphing parameter. We generated morphed images for three distinct values of ρ: 0.3 and 0.7, creating 100 images for each value of ρ. Examples of morphed images with varying ρ are shown in Fig. 2

We constructed three MNIST-based datasets with different synthetic images, distinguished by the morphing parameter ρ. The naming convention for these datasets directly

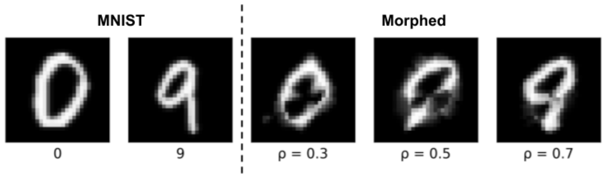


Figure 2. Examples of 0 and 9s from MNIST dataset and morphed digits generated, varying ρ.

represents the ρ values used for synthetic images corresponding to the digits 0 and 9, respectively.

1. **MNIST_0_1**: This dataset includes synthetic images that are selected from real images of their respective classes. It contains 300 synthetic images each for digits 0 and 9, corresponding to $\rho = 0.0$ for class 0 and $\rho = 1.0$ for class 9, representing ideal support images.
2. **MNIST_1_0**: In this dataset, 300 synthetic images for each class are counter-replicated: the images meant for class 0 are exact images of 9 ($\rho = 1.0$) and those for class 9 are exact images of 0 ($\rho = 0.0$), thus creating poor support images. In this case, the "synthetic" images are sampled from the real images of 0s and 9s, with no overlap with the real image classes.
3. **MNIST_0.7_0.3**: The third dataset includes synthetic images where for class 0, the images are morphs $M(\rho)$ with $\rho = 0.7$, and for class 9, $\rho = 0.3$. These morphed images blend characteristics of the two classes, presenting an intermediate case between perfect and counter-replicated images.

We generated k-shot datasets by randomly sampling k training images from each class. We used released train/validation/test splits for Flowers10 and CUBS10 and used a 0.8/0.1/0.1 split respectively randomly sampled for the MNIST datasets. In addition, for each experiment testing with synthetic images, we sample $\frac{k}{2}$ synthetic images for each class. We conducted experiments for values of $k = 8, 16, 32$ for Flowers10 and CUBS10 and used values of $k = 16, 32, 64$ for the MNIST dataset experiments.

4.2. Implementation

We use Bayesian optimization to optimize our class weighting hyperparameters. This process involves calling an objective function that is a nested training process with a contrastive layer and subsequently, a classifier layer. We set $n\_iter = 100$ and $n\_init = 20$ i.e., 20 evaluations of the objective function were conducted with randomly chosen hyperparameters before starting the Bayesian optimization.

Contrastive Layer Training In our contrastive layer training, we employ ResNet-18 as our baseline encoder network [8]. For each training iteration, we resample the training data, comprising both real and synthetic images. The network is trained for 200 epochs with an Adam optimizer, a learning rate of 0.001, batch size of 32, momentum of 0.9, temperature of 0.07, and weight decay of $1e-4$ [35].

Classifier Training After we train the contrastive layer, we freeze the embeddings learned and finetune the classifier. Here, the training data is resampled from a held-out dataset for each run of the classifier. Specifically, we select 16 images per class for all datasets. The classifier, a 3-layer MLP network, is trained with cross-entropy loss, a learning rate of 0.001, a batch size of 32, and 200 epochs.

Validation Set Consistency Throughout the experiment, we maintain a fixed validation set to evaluate model performance. For the Flowers10 dataset, we use the published train/val/test split, which comprises 10 validation images per class. For the CUB10 dataset, we use 30 images per class. For each of the MNIST dataset experiments, we use 64 images per class for validation.

For each hyperparameter set, we train the model over $n_{\text{contr}} = 3$ contrastive runs, each involving $n_{\text{classif}} = 3$ classifier runs with different training data samples. We compute the average validation accuracy for each contrastive run from its n_{classif} classifiers. The overall performance for a set of hyperparameters is then the mean of these averages across the n_{contr} runs, involving $n_{\text{contr}} \times n_{\text{classif}} = 9$ classifier training. The final reported validation accuracy is this average, along with the standard deviation, across the 9 runs. This process is delineated in lines 4-14 of Algorithm 1.

4.3. Baseline Experiments

SupCon-Real: We train a supervised contrastive network for each dataset using the SupCon loss on only original images and their corresponding labels.
SupCon-Mixed: We train a supervised contrastive network for each dataset including the support set images for each class using the SupCon loss. In this case, the support set images were labeled as the same label as its corresponding class.

4.4. Manual Weight Testing

Furthermore, baseline testing was conducted where a uniform value of w was applied across all class-specific weights. Note that when $w_{y_i} = 0$ for all $y_i \in C$, L_{UniCon} behaves very similarly to $L_{SupCon-Real}$. On the other hand, when $w_{y_i} = 1$ for all $y_i \in C$, the UniCon loss fully considers all similarities between the embeddings of real and synthetic images in the overall loss for every anchor image that is a real image. In this case, $L_{SupCon-Mixed}$ behaves similarly but differs in considering both real and synthetic images as anchor images for each batch. These results are reported in the supplementary material.

4.5. UniCon Results

We report our results on Flowers10 and CUBS10 with DALL-E-generated synthetic images and Stable Diffusion-generated synthetic images. We compare UniCon to SupCon-Real and SupCon-Mixed. We differentiate between experiments using DALL-E synthetic images versus Stable Diffusion synthetic images with the notation SupCon-Mixed-D and UniCon-D, and SupCon-Mixed-S and UniCon-S. We report the learned weights returned by our UniCon method and the respective validation accuracy for each of the experiments and compare them to the baseline methods. Results are reported in Table 1

UniCon is effective in leveraging synthetic images from different sources, especially in the few-shot learning scenario with $k = 8$, as shown in Table 1. UniCon consistently outperforms the SupCon-Real baseline, which ignores synthetic images entirely, and the SupCon-Mixed baseline, which incorporates synthetic images without considering their quality.

For the Flowers10 dataset with $k = 8$, UniCon-D achieves an accuracy of 85.65%, surpassing SupCon-Real by 5.33% and SupCon-Mixed-D by 4.86%. In the CUBS10 dataset with $k = 8$, UniCon-D reaches an accuracy of 58.83%, surpassing SupCon-Real by 6.48%. We see similar trends with the UniCon experiments using Stable diffusion-generated synthetic images, showing robustness and adaptability to different synthetic image sources. These gains in the few-shot setting highlight UniCon's ability to effectively utilize synthetic data when real examples are scarce. Notably, in cases where SupCon-Mixed performs worse than SupCon-Real, such as for CUBS10 with DALL-E synthetics at $k = 8$ and $k = 16$, UniCon can mitigate the negative impact of lower-quality synthetics and achieve gains over both baselines.

Figure 3 provides a visual comparison between real and synthetic images generated by DALL-E and Stable Diffusion for three out of the ten classes from the Flowers10 and CUBS10 datasets, respectively. Each class row showcases two real images alongside two synthetic images from each generative model, with the average learned weights from UniCon displayed beneath the synthetic sets. We inspect the quality of these three classes per dataset across image types based on the generative model used and the corresponding weights learned.

For the Flowers10 dataset, the wallflower class shows synthetic images that are visually similar to the real ones, reflected in the relatively high learned weights - DALL-E (w=0.56) and Stable Diffusion (w=0.63)- indicating a stronger trust in the synthetically generated data for augmenting the learning process. The petunia class qualitatively demonstrates the generative models' struggle with color accuracy and pattern replication, which is particularly challenging for classes with a high degree of intra-class color variation. Thus, the learned weights are more moderate, suggesting that these images are less useful for learning accurate representations. The discrepancy is more pronounced for the watercress class, where Stable Diffusion images (w=0.08) are notably less realistic than DALL-E images (w=0.29), leading to a lower average learned weight for Stable Diffusion. The images show a significant deviation from the real data images, prompting minimal reliance on these synthetics for training. Interestingly, the Watercress class showcases an instance where UniCon with learned weights close to zero outperforms SupCon-Real. We believe that this result can be attributed to the fact that even low-quality synthetic images can serve as informative negative examples in contrastive learning. By down-weighting their contribution to the loss, UniCon effectively leverages these examples to shape the representation space without allowing them to dominate the learning process. In contrast, SupCon-Real completely discards this information.

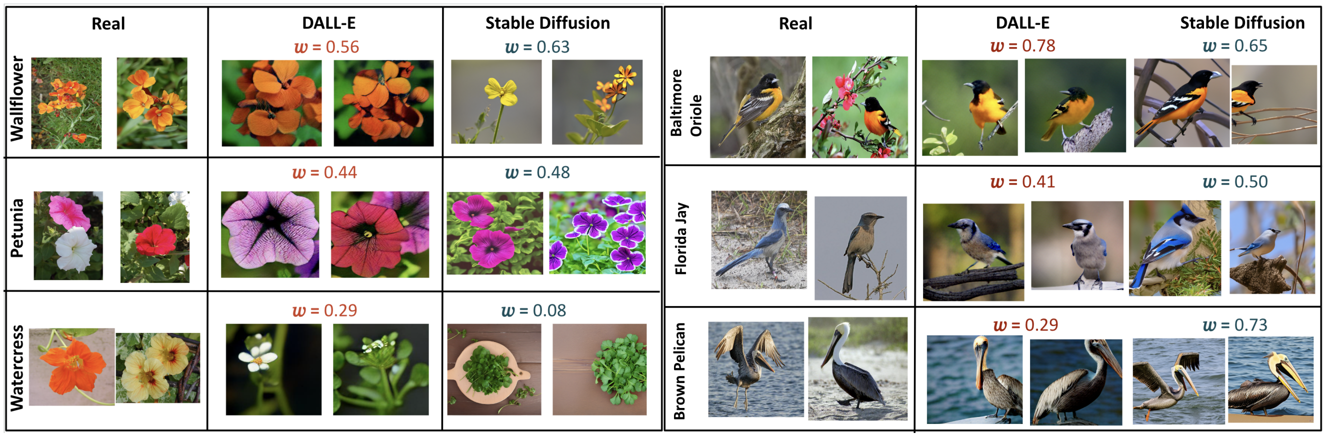
Subfigure (b) of Figure 3 focuses on classes from the CUBS10 dataset, specifically Baltimore oriole, Florida jay, and brown pelican. For the Baltimore oriole class, DALL-E (w=0.78) and Stable Diffusion (w=0.65) both generate relatively realistic images, capturing the essential characteristics of the species. Similarly, the Florida Jay class shows comparable image quality between DALL-E (w=0.41) and Stable Diffusion (w=0.50). However, the brown pelican class reveals a notable difference, with Stable Diffusion images (w=0.73) appearing more realistic and better capturing the distinctive features of the species compared to DALL-E images (w=0.29), corresponding to a higher average learned weight for Stable Diffusion.

The learned weights not only seem to reflect the quality and relevance of the synthetic images but also play a crucial role in building better representations of the real images for downstream classification tasks. By assigning higher weights to informative and reliable synthetic examples and lower weights to noisy or misleading ones, UniCon effectively guides the contrastive learning process to focus on the most relevant features and relationships present in the real data. This selective emphasis on high-quality synthetic data helps to construct more robust and discriminative representations of the real images, ultimately leading to improved classification performance. The supplementary material includes detailed reports on the learned weights learned by the UniCon experiments across all classes, datasets, and synthetic image sources.

4.6. MNIST Studies

We conducted a series of experiments to validate the effectiveness of UniCon with synthetic images of varying uncertainty levels. In these studies, we controlled the degree of synthetic data relevance by setting ρ during the morph image generation. In this setup, we then applied the UniCon

Figure 3. **Comparative visualization of real and synthetic images from UniCon experiments** generated by DALL-E and Stable Diffusion for selected classes from the Flowers10 and CUBS10 datasets. The weights displayed below the synthetic images represent the average learned weights learned by UniCon for each class across all k-shot experiments. The UniCon weights correlate with the assessed utility of these images in enhancing the model's training efficacy for fine-grained image classification.



(a) Flowers10 dataset: Real vs. synthetic images of Wallflower, Petunia, and Watercress with average learned UniCon weights indicated for DALL-E and Stable Diffusion.

(b) CUBS10 dataset: Real vs. synthetic images of Baltimore Oriole, Florida Jay, and Brown Pelican with average learned UniCon weights indicated for DALL-E and Stable Diffusion.

Table 1. **Fine-Grained Classification Performance for Flowers10 and CUBS10** UniCon with the best weighting outperforms both SupCon-Real and SupCon-Mixed, in classification accuracy across all k for both types of synthetic images. The average validation accuracy and corresponding standard deviation for all experiments are reported.

	Flowers10			CUBS10		
k	8	16	32	8	16	32
SupCon-Real	80.32 (3.05)	86.34 (2.79)	92.36 (2.78)	52.35 (2.35)	61.23 (1.57)	68.87 (2.38)
SupCon-Mixed-D	80.79 (3.87)	87.27 (2.44)	90.86 (1.76)	58.76 (2.76)	64.58 (2.65)	70.49 (2.07)
UniCon-D	**85.65 (2.12)**	**90.16 (0.11)**	**93.40 (1.04)**	**58.83 (0.33)**	**65.47 (0.66)**	**71.60 (0.72)**
SupCon-Mixed-S	81.94 (1.30)	86.69 (2.63)	91.32 (1.70)	55.94 (2.71)	63.43 (1.73)	70.06 (1.03)
UniCon-S	**84.03 (0.56)**	**90.51 (0.16)**	**92.71 (1.24)**	**56.79 (2.34)**	**66.32 (0.59)**	**71.14 (1.01)**

method to learn the weights corresponding to each synthetic image class. Given our prior understanding and control over the morphing in the synthetic set, we had an a priori notion of the learned weighting w for optimizing the accuracy of UniCon in handling synthetic images. Subsequently, we conducted experiments to verify our predictions that UniCon would 1) learn weights that correspond to the relevance of the synthetic data and 2) achieve higher accuracy using the learned weights.

We conducted three experiments to this end using the aforementioned MNIST-derived datasets: **MNIST_0_1**, **MNIST_1_0**, and **MNIST_0.7_0.3**. Results for the former two datasets are reported in Table 2 and for the latter dataset in Table 3.

For the **MNIST_0_1** dataset, where the synthetic images were selected from the real images($\rho = 0$ for class 0, $\rho = 1$ for class 9), the expected learned weights should be close to 1 for both classes. The UniCon method, through Bayesian optimization, correctly identified and returned these expected learned weights - $[0.75, 0.97]$ for $k = 16$, $[1.0, 1.0]$ for $k = 32$, and $[0.7, 0.87]$ for $k = 64$. These high-weight values indicate that UniCon recognized the high quality and reliability of the synthetic images, appropriately weighting them almost equally to the real images in the contrastive loss calculation.

On the other hand, for the **MNIST_1_0** dataset, the synthetic images were selected from real images from the opposite class, for class 0 were replicas of class 9, and vice versa. These highly untrustworthy and misleading synthetic images required learned weights close to 0 to essentially disregard them during training. Again, UniCon with Bayesian optimization successfully identified the expected learned weights as $[0.0, 0.0]$ across all k values. These zero weights mean UniCon correctly recognized that the synthetic images were completely unreliable and should not contribute at all to the contrastive loss.

Table 2. **MNIST_0_1 and MNIST_1_0 Classification Performance** of UniCon Against SupCon-Real and SupCon-Mixed Methods. We report UniCon's performance and the learned weights $[w_0, w_9]$ for synthetic images corresponding to real image classes C_0 and C_9 respectively for all k-shot experiments for $k = 16, 32, 64$.

	MNIST_0_1			**MNIST_1_0**		
k	16	32	64	16	32	64
SupCon-Real	85.50 (2.92)	91.41 (3.33)	93.92 (2.08)	88.45 (4.03)	91.49 (3.65)	95.40 (2.43)
SupCon-Mixed	90.45 (3.49)	92.36 (2.80)	95.31 (1.47)	65.71 (4.25)	71.01 (2.67)	69.44 (5.17)
UniCon	**92.01 (2.44)**	**95.14 (1.88)**	**95.40 (2.21)**	**90.36 (4.11)**	**92.62 (1.01)**	**94.79 (0.95)**
Weights $[w_0, w_9]$	[0.75, 0.97]	[1.00, 1.00]	[0.70, 0.87]	[0.00, 0.00]	[0.00, 0.00]	[0.00, 0.00]

These results highlight UniCon's ability to automatically assign appropriate weights - high weights near 1.0 like $[0.75, 0.97]$ and $[1.0, 1.0]$ for trustworthy synthetics in **MNIST_0_1** to boost performance over SupCon-Real by up to 6.5% for $k = 16$. In stark contrast, for untrustworthy synthetics in **MNIST_1_0**, the negligible weights $[0.0, 0.0]$ enabled UniCon to disregard the misleading data and perform comparably (within 1-2%) to SupCon-Real, crucially avoiding the significant 20%+ drop suffered by SupCon-Mixed.

Table 3. **MNIST_0.7_0.3 Classification Performance** of UniCon Against SupCon-Real and SupCon-Mixed Methods. We report UniCon's performance and the learned weights $[w_0, w_9]$ for synthetic images corresponding to real image classes C_0 and C_9 respectively for all k-shot experiments for $k = 16, 32, 64$.

	MNIST_0.7_0.3		
k	16	32	64
SupCon-Real	87.59 (3.00)	92.19 (3.61)	94.01 (1.61)
SupCon-Mixed	89.15 (2.82)	91.06 (4.65)	93.84 (2.56)
UniCon	**92.53 (0.85)**	**93.57 (1.73)**	**95.83 (1.61)**
Weights $[w_0, w_9]$	[0.62, 0.38]	[0.76, 0.83]	[1.00, 0.91]

For the **MNIST_0.7_0.3** dataset with intermediate synthetic image uncertainty (morphed digits blending 0 and 9, with $\rho = 0.7$ for class 0 and $\rho = 0.3$ for class 9), the expected learned weights should lie between 1.0 (highly trustworthy) and 0.0 (untrustworthy). Specifically, the weight for class 0 synthetics ($\rho = 0.7$) should be lower than class 9 ($\rho = 0.3$) due to higher uncertainty.

The varying weights across different k values could potentially arise due to noise or variance in the data. With smaller values of k (e.g., k=16), the real examples might not sufficiently capture the true data distribution, leading to higher uncertainty. In such cases, UniCon should lower the weights of the synthetics to mitigate their influence. As k increases (e.g., k=64), the real examples likely provide a better representation of the data, reducing uncertainty. Consequently, UniCon can afford to assign higher weights to partially trustworthy synthetics, leveraging them to boost performance. Class complexity and intra-class variations could also influence the weight variations.

The learned weights exhibit the expected pattern based on the uncertainty levels of the two classes of synthetic images. This, coupled with the quantitative accuracy gains over baselines, validates UniCon's ability to automatically identify and appropriately weight synthetic images of varying uncertainty levels. This enables UniCon to effectively leverage partially trustworthy synthetic data while mitigating the negative impacts of highly uncertain samples.

5. Conclusion

In this work, we introduced Uncertainty-Inclusive Contrastive Learning (UniCon), a novel contrastive learning framework that incorporates uncertainty weights for synthetic images, allowing us to effectively learn from synthetic images with varying quality. UniCon showed consistent improvements in model performance on vision classification tasks across two fine-grained datasets, outperforming standard contrastive learning baselines both with and without synthetic images. The class-specific weights learned by UniCon match expectations of data relevance based on qualitative analysis. UniCon provides a principled and adaptable approach to leveraging synthetic data in representation learning, particularly in data-scarce domains.

In future work, we plan to explore more advanced optimization techniques to further improve the efficiency and scalability of the weight learning process. Additionally, we aim to extend our experimentation across a broader spectrum of domains and datasets in diverse real-world scenarios, investigating the potential of UniCon in handling various types of uncertainties and noise in synthetic data. This will help us better understand and address the challenges of using synthetic data in nuanced domains, where generative AI models may struggle to capture pertinent attributes. Overall, our findings underscore UniCon's potential as a valuable tool for effectively leveraging synthetic images in vision classification tasks, paving the way for more accurate and reliable models that incorporate synthetic data.

References

[1] Shekoofeh Azizi, Simon Kornblith, Chitwan Saharia, Mohammad Norouzi, and David J. Fleet. Synthetic data from diffusion models improves imagenet classification, 2023.

[2] Pierre Chambon, Christian Bluethgen, Curtis P Langlotz, and Akshay Chaudhari. Adapting pretrained vision-language foundational models to medical imaging domains. *arXiv preprint arXiv:2210.04133*, 2022.

[3] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[4] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[5] Lijie Fan, Kaifeng Chen, Dilip Krishnan, Dina Katabi, Phillip Isola, and Yonglong Tian. Scaling laws of synthetic images for model training ... for now, 2023.

[6] Peter I. Frazier. A tutorial on bayesian optimization, 2018.

[7] Maayan Frid-Adar, Idit Diamant, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. Gan-based synthetic medical image augmentation for increased cnn performance in liver lesion classification. *Neurocomputing*, 321:321–331, 2018.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[9] Ruifei He, Shuyang Sun, Xin Yu, Chuhui Xue, Wenqing Zhang, Philip Torr, Song Bai, and Xiaojuan Qi. Is synthetic data from generative models ready for image recognition?, 2023.

[10] Tim Head, Manoj Kumar, Holger Nahrstaedt, Gilles Louppe, and Iaroslav Shcherbatyi. scikit-optimize/scikit-optimize, 2021.

[11] Reyhane Askari Hemmat, Mohammad Pezeshki, Florian Bordes, Michal Drozdzal, and Adriana Romero-Soriano. Feedback-guided data synthesis for imbalanced classification, 2023.

[12] Ali Jahanian, Xavier Puig, Yonglong Tian, and Phillip Isola. Generative models as a data source for multiview representation learning, 2022.

[13] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels, 2018.

[14] Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling, 2019.

[15] Gary Marcus, Ernest Davis, and Scott Aaronson. A very preliminary analysis of dall-e 2, 2022.

[16] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[17] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.

[18] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation, 2021.

[19] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2): 3, 2022.

[20] Haroon Rashid, M Asjid Tanveer, and Hassan Aqeel Khan. Skin lesion classification using gan based data augmentation. In *2019 41St annual international conference of the IEEE engineering in medicine and biology society (EMBC)*, pages 916–919. IEEE, 2019.

[21] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning, 2019.

[22] Zhongzheng Ren and Yong Jae Lee. Cross-domain self-supervised multi-task feature learning using synthetic imagery, 2017.

[23] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[24] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation, 2023.

[25] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

[26] Mert Bulent Sariyildiz, Karteek Alahari, Diane Larlus, and Yannis Kalantidis. Fake it till you make it: Learning transferable representations from synthetic imagenet clones, 2023.

[27] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms, 2012.

[28] Yonglong Tian, Lijie Fan, Phillip Isola, Huiwen Chang, and Dilip Krishnan. Stablerep: Synthetic images from text-to-image models make strong visual representation learners, 2023.

[29] Satoshi Tsutsui, Yanwei Fu, and David Crandall. Reinforcing generated images via meta-learning for one-shot fine-grained visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(3):1455–1463, 2024.

[30] Vishaal Udandarao, Ankush Gupta, and Samuel Albanie. Sus-x: Training-free name-only transfer of vision-language models, 2023.

[31] Boris van Breugel and Mihaela van der Schaar. Beyond privacy: Navigating the opportunities and challenges of synthetic data, 2023.

[32] Boris van Breugel, Zhaozhi Qian, and Mihaela van der Schaar. Synthetic data, real errors: how (not) to publish and use synthetic data, 2023.

[33] Roy Voetman, Maya Aghaei, and Klaas Dijkstra. The big data myth: Using diffusion models for dataset generation to train deep detection models, 2023.

[34] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. Cub-200-2011, 2022.

[35] Feng Wang and Huaping Liu. Understanding the behaviour of contrastive loss, 2021.

[36] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. Git: A generative image-to-text transformer for vision and language. *arXiv preprint arXiv:2205.14100*, 2022.

[37] Yifei Wang, Jizhe Zhang, and Yisen Wang. Do generated data always help contrastive learning?, 2024.

[38] Shin'ya Yamaguchi and Takuma Fukuda. On the limitation of diffusion models for synthesizing training datasets, 2023.

[39] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2(3):5, 2022.

[40] Chenshuang Zhang, Chaoning Zhang, Mengchun Zhang, and In So Kweon. Text-to-image diffusion model in generative ai: A survey. *arXiv preprint arXiv:2303.07909*, 2023.

[41] Renrui Zhang, Xiangfei Hu, Bohao Li, Siyuan Huang, Hanqiu Deng, Hongsheng Li, Yu Qiao, and Peng Gao. Prompt, generate, then cache: Cascade of foundation models makes strong few-shot learners, 2023.