Zero-Shot Learning on AWA2 Dataset Image Classification

Jizheng Chen 520021911182

May 12, 2023

Abstract

This report presents an implementation of machine learning models for the task of zero-shot image classification on the Animals with Attributes 2 (AWA2) dataset. It covers three main models: Semantic Relatedness, Semantic Embedding, and Synthetic Dataset Generation. All the models are implemented using the PyTorch library.

1 Introduction

Zero-Shot Learning (ZSL) is a machine learning paradigm where the model is trained to recognize objects from classes not seen during the training phase. In this report, we focus on zero-shot image classification, particularly on the Animals with Attributes 2 (AWA2) dataset. The models implemented are Semantic Embedding (SemEmb), Synthetic Dataset Generation (SynTrainer), and Semantic Relatedness (SemRel), each playing a crucial role in the overall zero-shot learning framework.

- **Semantic Embedding (SemEmb)**: In the context of zero-shot learning, SemEmb learns a high-dimensional embedding for each class of images, capturing the intrinsic characteristics of the class. The embeddings are learned in a way that semantically similar classes are closer in the high-dimensional space.
- **Synthetic Dataset Generation (SynTrainer)**:SynTrainer uses a Generative Adversarial Network (GAN) to generate synthetic data. The synthetic data aims to bridge the gap between seen and unseen classes, allowing the model to generalize better to unseen classes.
- **Semantic Relatedness (SemRel)**:SemRel assesses the degree of relatedness between different classes. This is achieved by using a trained model to generate embeddings for the classes, and then measuring the distance between these embeddings in the high-dimensional space. This step is essential to determine the relationship between seen and unseen classes.

2 Implementation Details

Each of the three models (SemEmb, SynTrainer, and SemRel) is encapsulated into a Python class that can be trained and evaluated. The models are implemented using PyTorch.

2.1 Semantic Relatedness Method

The experiment uses a class SemRel(), which is an implementation of a classifier based on semantic relatedness. The class is initialized with training and testing datasets, a model, a criterion for loss calculation (CrossEntropyLoss in this case), an optimizer (Adam optimizer), and a similarity matrix. The datasets used are trainvalclasses.txt for training and testclasses.txt for testing. The model is moved to the GPU if available, else it runs on CPU.

The SemRelClassifier model is trained using the Adam optimizer with a learning rate passed from the args. The model's parameters are updated in the training process to minimize the cross entropy loss.

A significant element in this approach is the similarity matrix, which is computed from the predicate binary matrix of the training dataset. This matrix captures the semantic relatedness between different

classes in the dataset. Cosine similarity is used to measure the relatedness between different class predicates. The similarity matrix is then normalized.

The training process involves running the model for a specified number of epochs. In each epoch, the model's parameters are updated to minimize the loss. The loss and accuracy of the model on the training dataset are logged after each epoch. The model is also evaluated on the testing dataset at certain intervals (args.eval_iter).

The evaluation process involves passing the test data through the model and comparing the predicted classes with the true classes. The output of the model is multiplied with the similarity matrix and softmax operation is applied to get a probability distribution over the classes. The predicted class is determined by selecting the class with maximum probability. The mean accuracy of the model is calculated by comparing the predicted classes with the true classes.

2.2 Semantic Embedding Method

The Semantic Embedding approach is implemented in the SemEmb() class. Similar to the previous method, this class is initialized with training and testing datasets, a model, a criterion for loss calculation (in this case, Binary Cross Entropy Loss), an optimizer (Adam optimizer), and it either runs on the GPU, if available, or CPU.

The model, Classifier(), is trained with the Adam optimizer with a learning rate passed from the args. The model's parameters are updated in the training process to minimize the binary cross entropy loss.

The training process runs the model for a specified number of epochs. In each epoch, the model's parameters are updated to minimize the loss. The accuracy of the model on the training dataset is logged after each epoch. The model is also evaluated on the testing dataset at certain intervals (args.eval_iter).

The key difference in the Semantic Embedding approach is in how it predicts the class labels. The predicted labels are obtained by passing the data through the model. For each predicted label, it computes the Euclidean distance to the class labels in the predicate binary matrix. The class with the smallest distance is selected as the predicted class.

The evaluation process involves passing the test data through the model and comparing the predicted classes with the true classes. The mean accuracy of the model is calculated by comparing the predicted classes with the true classes.

2.3 Synthetic Method

The SynTrainer class is the core of the synthetic method. It is initialized with parameters that configure the model architecture, datasets, and training settings. This class has numerous methods to perform various tasks such as fitting the classifier, fitting the GAN (Generative Adversarial Network), fitting the final classifier, creating a synthetic dataset, and training the whole system.

The GAN consists of a generator and a discriminator. The generator creates synthetic data samples, and the discriminator judges whether a given sample is real or generated by the generator. The GAN is trained through a two-player min-max game where the generator tries to fool the discriminator, and the discriminator tries to correctly classify real and synthetic samples. The generator and discriminator are trained alternately: the discriminator is trained several times within each training loop, while the generator is trained once.

The synthetic method also involves a pre-training phase where a classifier is trained on the real training set. If the pre-trained model exists, it is loaded; otherwise, the model is trained and saved. After pre-training, the GAN is trained using the classifier's loss to guide the generator. The GAN's generator and discriminator models are saved after training.

Synthetic data samples are then created using the trained generator. These samples, together with the real training data, are used to train the final classifier. This process leverages the synthetic data to learn a more robust and generalized representation, reducing the risk of overfitting to the real training data.

The synthetic method is then evaluated by comparing the predicted class labels against the true labels. The average accuracy over all samples is calculated and reported for both the training and test data. The evaluation is performed at regular intervals during the training of the final classifier to monitor the progress and performance of the model.

This methodology demonstrates the effectiveness of synthetic data in improving the performance of the model. The synthetic data samples provide additional information and diversity, enabling the model to generalize better to unseen data. The method also ensures that the synthetic data is not just random noise, but meaningful samples that resemble the real data distribution, as enforced by the GAN. This approach might be particularly beneficial when dealing with imbalanced datasets, rare events, or when there is a shortage of labeled data.

3 Experiment Results

Table 1 presents the accuracy results of the semantic relatedness model with different learning rates for two versions of the model: small and big.

Overall, the big model outperforms the small model across all tested learning rates. This suggests that the big model has a better capacity to learn and generalize the task of semantic relatedness, likely due to its larger number of parameters that can capture more complex patterns in the data.

For both models, a learning rate of 5e-4 yields the best results. The accuracy decreases as the learning rate is reduced. This trend indicates that a higher learning rate enables faster convergence to a better solution in the model's parameter space. However, it's important to note that a learning rate that is too high may cause instability in the training and prevent the model from converging to a good solution.

Figure 3 shows the accuracy of semantic relatedness models.

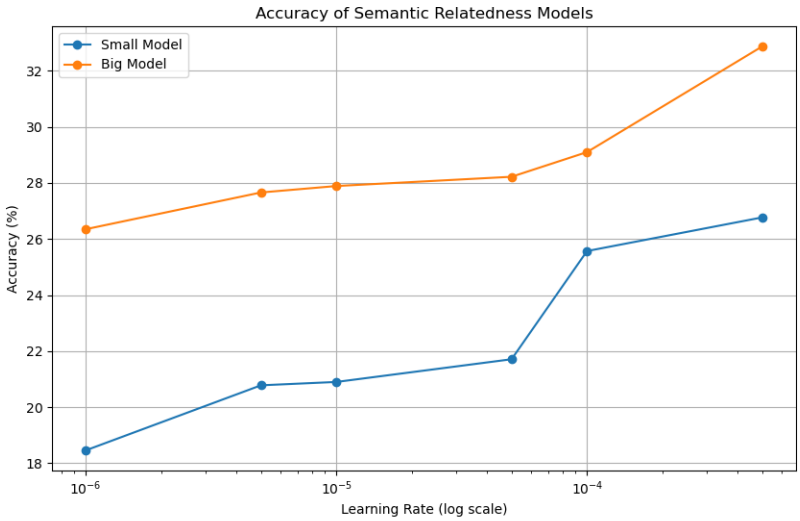

Figure 1: Supplementary Backup Data Construction

Table 2 presents the accuracy results of the semantic embedding model with different learning rates for two versions of the model: small and big.

Figure 2 shows the accuracy of semantic embedding models.

- **Small Model**: This model consists of a single hidden layer and uses ReLU (Rectified Linear Unit) as the activation function. The input dimension is specified by in_dim (2048 by default) and the hidden layer has a dimension of hidden_dim (128 by default). This hidden layer is followed by batch normalization and dropout with a rate of 0.5 for regularization, and finally a linear layer to project to num_labels (85 by default) output classes. The final activation function is a Sigmoid function, which scales the output to the range [0,1], which can be useful for binary or multi-label classification tasks.

- **Big Model**: This model has three hidden layers with dimensions 2000, 1200, and 1200 respectively. The activation function used in this model is LeakyReLU, which is a variant of ReLU that allows small negative values when the input is less than zero, providing a way to mitigate the

Learning Rate	Model	Accuracy
5e-4	small	26.77
1e-4	small	25.57
5e-5	small	21.71
1e-5	small	20.90
5e-6	small	20.78
1e-6	small	18.46
5e-4	big	32.87
1e-4	big	29.10
5e-5	big	28.22
1e-5	big	27.89
5e-6	big	27.66
1e-6	big	26.35

Table 1: Accuracy results for the semantic_relatedness models

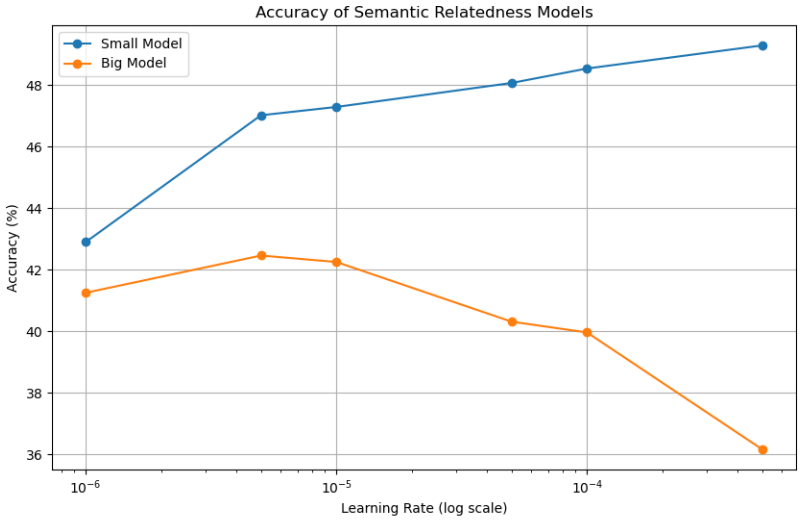

Figure 2: Accuracy of semantic embedding models

"dying ReLU" problem where some neurons can become inactive and stop learning. Each hidden layer is followed by dropout with a rate of 0.2 for regularization. The final layer projects to num_labels (85 by default) output classes, and a Sigmoid function is used as the final activation function, just like in the small model.

Interestingly, for this method, the small model outperforms the big model across all tested learning rates. This is in contrast with the results observed for the semantic relatedness model. It could suggest that for the task of semantic embedding, the small model is sufficient to capture the complexity of the data, and the additional parameters in the big model may lead to overfitting or do not provide additional benefit.

For the small model, the learning rate of 5e-4 yields the best accuracy, and the performance decreases as the learning rate is reduced. This is consistent with the trend observed in the semantic relatedness model.

For the big model, the highest accuracy is achieved with a learning rate of 5e-6. This indicates that for the big model, a smaller learning rate may be more suitable, possibly because a slower learning process helps prevent overfitting and allows the model to better generalize the data patterns.

Table 3 presents the results of the synthetic model with a complex grid of hyperparameters, including learning rate, generator learning rate (g_lr), discriminator learning rate (d_lr), and number of samples. Figure 3 shows the accuracy of synthetic models for different learning rates and sample

Learning Rate	Model	Accuracy
5e-4	small	49.29
1e-4	small	48.54
5e-5	small	48.07
1e-5	small	47.29
5e-6	small	47.02
1e-6	small	42.90
5e-4	big	36.16
1e-4	big	39.96
5e-5	big	40.31
1e-5	big	42.25
5e-6	big	42.46
1e-6	big	41.25

Table 2: Accuracy results for the semantic_embedding models

number . Here, we observe several patterns:

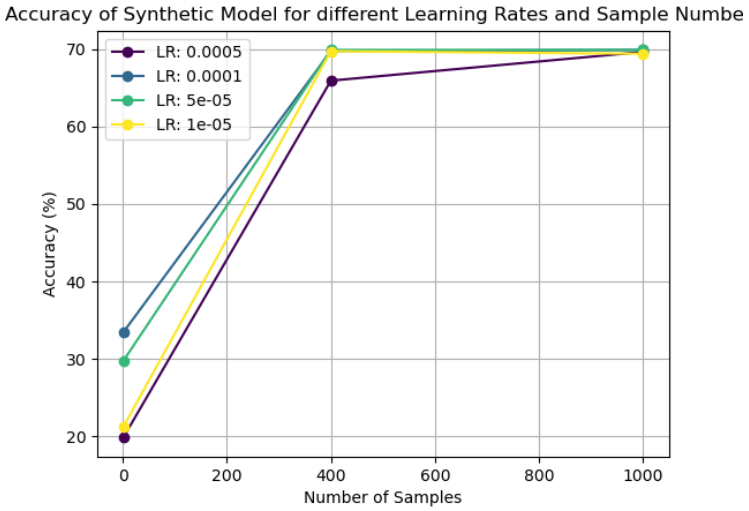

Figure 3: Accuracy of synthetic models

- **Number of Samples**: In nearly all cases, increasing the number of samples from 1 to 400 or 1000 results in a significant improvement in accuracy, implying that the synthetic model benefits from more data for training. This is understandable as more samples provide the model with more information to learn the underlying data distribution.
- **Learning Rate**:There is no clear trend across different learning rates. However, lower learning rates such as 1e-5 and 5e-5 seem to perform better in most cases. This may be because a lower learning rate allows the model to converge more precisely to a good solution.
- **Generator and Discriminator Learning Rates (g_lr and d_lr)**:A smaller generator learning rate (1e-6) generally performs better than a larger one (1e-5). The discriminator learning rate does not show a clear trend, but for most configurations, a rate of 1e-4 leads to higher accuracy.

4 Conclusion

In the conducted experiments, we evaluated three different methods for zero-shot image classification on the AWA2 dataset: semantic relatedness, semantic embedding, and synthetic methods.

Semantic Relatedness: This model showed a moderate level of accuracy with the best performance recorded for the 'big' model at a learning rate of 5e-4, reaching an accuracy of 32.87%. The smaller model consistently underperformed in comparison to the big model at all tested learning rates.

Semantic Embedding: This model outperformed the semantic relatedness model across all tested conditions. The 'small' model demonstrated a peak accuracy of 49.29% at a learning rate of 5e-4. Interestingly, the 'big' model achieved a maximum accuracy of 42.46

Synthetic Model: This model showed the highest performance among all methods, with the top accuracy reaching nearly 70%. The model's performance was strongly influenced by the number of samples, with more samples generally leading to better accuracy. The learning rates for both the overall model and the generator and discriminator components also played a significant role, with lower rates often leading to better results. However, the optimal values for these hyperparameters were not consistent across different conditions, highlighting the need for careful tuning.

In conclusion, while all methods demonstrated some capability for zero-shot image classification, the synthetic model showed the most promising results on the AWA2 dataset. The significant improvement in accuracy with the synthetic model suggests that it might be better suited for this task, possibly due to its ability to generate artificial data points, allowing it to handle the 'zero-shot' aspect of the task more effectively. Future research could explore more sophisticated versions of the synthetic model or different ways to combine the strengths of the three methods to further enhance performance.

Learning Rate	g_lr	d_lr	Num Samples	Accuracy
5e-4	1e-6	1e-5	1	19.85
5e-4	1e-6	1e-5	400	65.91
5e-4	1e-6	1e-5	1000	69.62
5e-4	1e-6	1e-4	1	17.21
5e-4	1e-6	1e-4	400	69.86
5e-4	1e-6	1e-4	1000	69.96
5e-4	1e-5	1e-5	1	19.56
5e-4	1e-5	1e-5	400	69.89
5e-4	1e-5	1e-5	1000	69.34
5e-4	1e-5	1e-4	1	18.82
5e-4	1e-5	1e-4	400	65.91
5e-4	1e-5	1e-4	1000	68.91
1e-4	1e-6	1e-5	1	33.44
1e-4	1e-6	1e-5	400	69.72
1e-4	1e-6	1e-5	1000	69.87
1e-4	1e-6	1e-4	1	31.76
1e-4	1e-6	1e-4	400	69.70
1e-4	1e-6	1e-4	1000	68.12
1e-4	1e-5	1e-5	1	33.94
1e-4	1e-5	1e-5	400	69.47
1e-4	1e-5	1e-5	1000	63.54
1e-4	1e-5	1e-4	1	12.38
1e-4	1e-5	1e-4	400	63.68
1e-4	1e-5	1e-4	1000	69.76
5e-5	1e-6	1e-5	1	29.74
5e-5	1e-6	1e-5	400	69.87
5e-5	1e-6	1e-5	1000	69.88
5e-5	1e-6	1e-4	1	10.21
5e-5	1e-6	1e-4	400	69.90
5e-5	1e-6	1e-4	1000	69.76
5e-5	1e-5	1e-5	1	17.29
5e-5	1e-5	1e-5	400	62.98
5e-5	1e-5	1e-5	1000	69.80
5e-5	1e-5	1e-4	1	5.89
5e-5	1e-5	1e-4	400	65.31
5e-5	1e-5	1e-4	1000	69.17
1e-5	1e-6	1e-5	1	21.22
1e-5	1e-6	1e-5	400	69.71
1e-5	1e-6	1e-5	1000	69.38
1e-5	1e-6	1e-4	1	10.64
1e-5	1e-6	1e-4	400	68.89
1e-5	1e-6	1e-4	1000	69.61
1e-5	1e-5	1e-5	1	43.42
1e-5	1e-5	1e-5	400	68.04
1e-5	1e-5	1e-5	1000	69.41
1e-5	1e-5	1e-4	1	19.9
1e-5	1e-5	1e-4	400	61.72
1e-5	1e-5	1e-4	1000	67.31

Table 3: Accuracy results for the synthetic model