Technical Report - Assignment 2
Advanced artificial intelligence techniques

Adrian Dinu Urse

Task 1 – Classification with missing labels

Step 1: Initial Supervised Fine-Tuning

Objective: To establish a baseline performance using labeled images only.
Approach:
- **Model**: Pretrained ResNet50
- **Optimizer**: AdamW with an initial learning rate of 0.001.
- **Loss Function**: CrossEntropyLoss for multi-class classification.
- **Data Augmentation**: Standard augmentations, including random flips, and normalization.
- **Training Configuration**:
    - Number of Epochs: 20
    - Batch Size: 32

Results:
- Achieved a Kaggle score of **0.53**.
- Observations:
    - The pretrained ResNet50 adapted well to the labeled dataset, but the small size of labeled data limited further improvements.

Step 2: Semi-Supervised Fine-Tuning with FixMatch
Objective: To leverage the unlabeled images by applying the FixMatch algorithm, which combines pseudo-labeling and consistency regularization.
FixMatch Implementation:
- **Concept**:
    - FixMatch assigns pseudo-labels to unlabeled data using confident predictions from the model.
    - These pseudo-labels are treated as ground truth during training, but only if the confidence of the prediction exceeds a dynamic threshold.
- **Key Components**:
    - Weak Augmentation: Simple transformations for generating pseudo-labels.
    - Strong Augmentation: Complex transformations for enforcing consistency.

Configuration Details:
- **Scheduler**: CosineAnnealingLR with T_max=10 epochs to manage the learning rate decay.
- **Confidence Threshold**:
    - Initial Threshold: 0.70

- Final Threshold: 0.95
- **Schedule**: A linear interpolation between the initial and final thresholds:
threshold_schedule = lambda epoch: initial_threshold + (final_threshold - initial_threshold) * (epoch / num_epochs)

- **Unsupervised Loss Weight**:
  - Dynamically scaled using the schedule:
  lambda_u_schedule = lambda epoch: min(0.5, epoch / 10)
- **Training**:
  - Number of Epochs: 10
  - Batch Size: 64
  - Combined labeled and pseudo-labeled data.

Hyperparameter Impact

1. **Threshold Scheduling**:
   - Lower thresholds (e.g., 0.70) introduced noisy labels early in training.
   - Gradual increase to 0.95 ensured high-quality pseudo-labels in later epochs, leading to performance stability.
2. **Learning Rate Scheduler**:
   - CosineAnnealingLR allowed smooth decay, avoiding abrupt changes that could destabilize training.
3. **Unsupervised Loss Weight**:
   - A ramp-up schedule for the unsupervised loss weight ensured the model gradually adapted to pseudo-labeled data.

Results:

- Achieved a Kaggle score of **0.61**.
- Observations:
  - Adding pseudo-labeled data significantly improved the model's generalization.
  - The confidence threshold schedule played a key role in balancing precision and recall during pseudo-label generation.
  - CosineAnnealingLR effectively managed learning rate decay, preventing overfitting during the fine-tuning process.

Step 3: Fine-Tuning Vision Transformer (ViT)

Objective: To leverage transformer-based architectures for improved feature representation and performance.

Approach:

- **Model**: Vision Transformer (ViT), specifically google/vit-base-patch16-224-in21k, pretrained on ImageNet-21k.
- **Optimizer**: AdamW with an initial learning rate of 5e-5.
- **Scheduler**: StepLR with the following configuration:

    - **Step Size**: 3 epochs
    - **Gamma**: 0.1 (decay factor)
- **Loss Function**: CrossEntropyLoss for multi-class classification.

Training Configuration:

- Number of Epochs: 10
- Batch Size: 32
- Dataset: Labeled images only.

Hyperparameter Impact

1. **Threshold Scheduling**:
    - Lower thresholds (e.g., 0.70) introduced noisy labels early in training.
    - Gradual increase to 0.95 ensured high-quality pseudo-labels in later epochs, leading to performance stability.
2. **Learning Rate Scheduler**:
    - StepLR scheduler provided a structured learning rate decay that aligned well with the ViT's training dynamics.
3. **Unsupervised Loss Weight**:
    - A ramp-up schedule for the unsupervised loss weight ensured the model gradually adapted to pseudo-labeled data.

Results:

- Achieved a Kaggle score of **0.91**.
- Observations:
    - Fine-tuning ViT significantly outperformed ResNet50, showcasing the strength of transformer-based architectures for this task.

Applying FixMatch to ViT:

- After achieving a score of **0.91** using labeled data, FixMatch was applied to integrate unlabeled data:
    - Confidence Thresholds: Initial: 0.60, Final: 0.95
    - Scheduler: OneCycleLR
- **Results:**
    - Performance did not increase beyond **0.91**.
    - **Analysis**:
        - The high performance of ViT on labeled data left limited room for improvement with semi-supervised learning.
        - FixMatch's pseudo-labeling may have introduced noisy labels that hindered further improvement.

Step 4: Ensemble Using Fine-Tuned ViT and FixMatch ViT

Objective: To combine the predictions of the fine-tuned ViT and FixMatch ViT models to enhance performance.

Approach:

- **Weighted Averaging**: Predictions from both models were combined using a weighted average:
    - Fine-Tuned ViT weight: 0.7
    - FixMatch ViT weight: 0.3
- **Fallback Strategy**:
    - When the two models disagreed (different predicted classes), a confidence-based fallback was implemented:
        - If the average confidence across both models was below a threshold (0.7), the model with higher confidence was chosen.
        - Otherwise, the ensemble prediction was selected.

Results:

- Kaggle score: **0.91**
- Observations:
    - The ensemble did not improve upon the fine-tuned ViT's score of **0.91**.
    - Likely cause: Both models were trained on the same labeled dataset, leading to similar decision boundaries and predictions.

Step 5: Pseudo-Labeling with Fine-Tuned ViT

Objective: To leverage the fine-tuned ViT to generate pseudo-labels for unlabeled data and further train the model with an expanded dataset.

Approach:

- **Pseudo-Labeling Configuration:**
    - Initial Threshold: 0.90
    - Generated approximately 17,000 pseudo-labeled samples.
- **Training Configuration:**
    - Combined the labeled and pseudo-labeled datasets.
    - Fine-tuned the model for an additional 15 epochs.
    - **Optimizer**: AdamW with a reduced learning rate of 1e-6.

Results:

- Kaggle score: **0.91**
- Observations:
    - Despite expanding the training dataset, the model's performance did not improve.
    - Analysis suggests that the pseudo-labeled data may not have introduced sufficient new information to significantly alter the model's predictions.

Step 6: Test-Time Augmentation (TTA)

Objective: To apply test-time augmentations to improve the model's robustness during inference.

Approach:

- Applied TTA with multiple augmentations:
    - Horizontal flip
    - Small rotations (e.g., ±15 degrees)
    - Color jitter
- Averaged predictions across augmented versions of each test image.

Results:

- Kaggle score: **0.33**
- Observations:
    - TTA significantly degraded the model's performance.
    - Likely cause: The augmentations introduced distortions that the model was not trained to handle, resulting in poor predictions.

Task 2 – Classification with noisy labels

Step 1: Initial Fine-Tuning
Objective: To fine-tune a pretrained model on the noisy-labeled dataset and establish a baseline performance.
Approach:

- **Dataset Split**:
  - The provided dataset was split into training and testing subsets.
  - A stratified split was used to preserve label distributions in both subsets.
- **Model:** Pretrained ResNet50 from PyTorch's model zoo.
- **Criterion:**
  - CrossEntropyLoss with label smoothing to handle noise in labels
- **Optimizer**:
  - AdamW with a learning rate of 1e-4

Training Configuration:

- Number of Epochs: 15
- Batch Size: 32
- Data Augmentation: Applied standard augmentations, including random cropping, flipping, and normalization.

Results:

- Achieved a Kaggle score of **0.78**.

Step 2: DivideMix for Noisy Label Classification
Objective: To improve classification performance by leveraging DivideMix, a robust framework for training with noisy labels.
Approach:

- **Framework Overview:**
  - DivideMix trains two networks simultaneously by dividing the training data into clean and noisy subsets using probabilistic modeling.
  - Each network trains on the clean samples of the other network, encouraging robust learning and avoiding confirmation bias.

Results:

- Achieved a Kaggle score of **0.85**.

Observations:

- The DivideMix framework effectively handled noisy labels by leveraging co-training and probabilistic modeling.
- The co-divide strategy improved the robustness of both networks, leading to significant performance gains over the baseline.

Reproducibility Instructions

This project was developed using Google Colab, where all training processes are visible. To reproduce the results, follow these steps:

1. **Dataset Preparation:**
   - Upload the dataset's zip file into the content/data directory within your Colab environment.
   - Run the dataset extraction cell to prepare the data for training.
2. **Training for the Best Scores:**
   - For Task 1:
     - Execute the cells for Vision Transformer (ViT) fine-tuning.
   - For Task 2:
     - Execute the cells implementing the DivideMix framework.
3. **Dependencies and Environment:**
   - Ensure the required libraries are installed as per the Colab setup.
   - All code cells include the necessary configurations to train the models without additional setup.
4. **Running the Models:**
   - Execute the appropriate training cells step-by-step as outlined in the notebook.
   - The configurations and hyperparameters are pre-defined for optimal performance.

References

1. **ResNet:** He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep Residual Learning for Image Recognition*. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), 770-778. Paper Link
2. **FixMatch:** Sohn, K., Berthelot, D., Li, C.-L., Zhang, Z., Carlini, N., Cubuk, E. D., Kurakin, A., Zhang, H., & Raffel, C. (2020). *FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence*. Advances in Neural Information Processing Systems (NeurIPS). Paper Link
3. **Vision Transformer (ViT):** Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. International Conference on Learning Representations (ICLR). Paper Link
4. **DivideMix:** Li, J., Socher, R., & Hoi, S. (2020). *DivideMix: Learning with Noisy Labels as Semi-Supervised Learning*. International Conference on Learning Representations (ICLR). Paper Link