1615119-1403

1615119_c.docx

University

Document Details

Submission ID
trn:oid:::9123:922064847

Submission Date
Apr 9, 2025, 2:28 PM GMT+3

Download Date
Apr 9, 2025, 2:29 PM GMT+3

File Name
1615119_c.docx

File Size
459.9 KB

8 Pages

950 Words

6,237 Characters

0% detected as AI

The percentage indicates the combined amount of likely AI-generated text as well as likely AI-generated text that was also likely AI-paraphrased.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Detection Groups

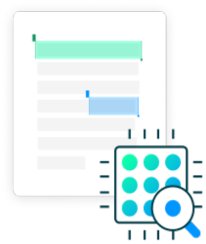
1 AI-generated only 0%
Likely AI-generated text from a large-language model.

2 AI-generated text that was AI-paraphrased 0%
Likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

Disclaimer
Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (it may misidentify writing that is likely AI generated as AI generated and AI paraphrased or likely AI generated and AI paraphrased writing as only AI generated) so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?
The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.



False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?
Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

Literature Survey

Problem Definitions and Learning Settings

The task of image classification, specifically cat vs. dog classification, falls under **supervised learning**, where input images are paired with labels. Key settings and challenges include:

Setting	Description
Supervised vs. Unsupervised	This project is supervised; labels are known. Unsupervised tasks such as clustering, lack labeled data.
Closed-set vs. Open-set	Closed-set: Only cats and dogs are expected. Open-set classification must handle unknown classes.
Domain shift	This project assumes no domain shift. In real scenarios, domain adaptation methods may be needed when train/test data distributions differ.

Challenges

- Intra-class variance (e.g., dog breeds look different)
- Inter-class similarity (e.g., furry cats and dogs)
- Small dataset size or class imbalance

Paper Survey and Trends

- Popular datasets: ImageNet, CIFAR-10, Stanford Dogs Dataset.
- ResNet and VGG are foundational CNNs from [He et al., 2015] and [Simonyan & Zisserman, 2014].
- Key keywords: "image classification", "transfer learning", "ResNet", "data augmentation".

Top venues: CVPR, ICCV, NeurIPS. Notable papers:

- **ResNet (He et al., 2015)**: Introduced residual connections to enable very deep networks.
- **EfficientNet (Tan & Le, 2019)**: Achieves SOTA accuracy with fewer parameters.

Recent Progress & Key Research Groups

- Facebook AI (Meta) and Google Brain are leading in computer vision.
- EfficientNet, ConvNeXt, and Vision Transformers are modern architectures.
- Vision Transformers (ViT, 2020) outperform CNNs at scale, but require more data and compute.

Baseline Method & Proposed Improvements

We selected **ResNet18** for its balance of accuracy and speed. Improvements could include:

- Fine-tuning all layers
- Adding dropout or batch normalization
- Using learning rate schedulers
- Trying architectures like EfficientNet-B0

Dataset and Preprocessing

Dataset

- Training set size: 2,000 images (1,000 cat + 1,000 dog)
- Validation set size: 500 images (250 cat + 250 dog)
- Test set: 500 unlabeled images

Preprocessing & Augmentation

- Resized to 128x128
- Normalized using [0.5, 0.5, 0.5]
- Augmentation:
  - RandomHorizontalFlip
  - RandomRotation(10 degrees)

Model Design

Pretrained **ResNet18**

- Backbone: Convolutional layers from ImageNet-pretrained ResNet18
- Final Layer Replaced: nn.Linear(512, 2)
- Loss Function: CrossEntropyLoss
- Optimizer: Adam (lr=0.001)
- Training: 5 epochs

- Input Image (3x128x128)
- ResNet18 Backbone (Frozen)
- Global Avg Pooling
- Linear Layer (512 → 2)
- Softmax / CrossEntropy
- Prediction (0 = Cat, 1 = Dog)

Training Strategy & Accuracy

- Trained on GPU (if available)
- Accuracy (Validation Set):
  Validation Accuracy (Val Acc) per Epoch:

```
Epoch [1/5] - Train Loss: 150.211 - Train Acc: 89.37%
       -> Val Loss: 29.225 - Val Acc: 92.26%

Epoch [2/5] - Train Loss: 147.951 - Train Acc: 89.62%
       -> Val Loss: 27.410 - Val Acc: 92.70%

Epoch [3/5] - Train Loss: 146.360 - Train Acc: 89.84%
       -> Val Loss: 27.873 - Val Acc: 92.14%

Epoch [4/5] - Train Loss: 145.399 - Train Acc: 89.88%
       -> Val Loss: 31.148 - Val Acc: 91.54%

Epoch [5/5] - Train Loss: 145.513 - Train Acc: 89.86%
       -> Val Loss: 28.207 - Val Acc: 92.32%
```

Epoch	Validation Accuracy
1	92.26%
2	92.70%
3	92.14%
4	91.54%
5	92.32%

During training, ResNet18 model achieved a peak validation accuracy of **92.70%** and a consistent training accuracy around **89.8%**, demonstrating good generalization. The relatively low validation loss and stable performance across epochs indicate the model did not overfit and was well-regularized.

Correct/Incorrect Predictions

```
import matplotlib.pyplot as plt
model.eval()

for images, image_ids in test_loader:
    outputs = model(images.to(device))
    _, preds = torch.max(outputs, 1)

    for i in range(3):  # Show 3 examples
        img = images[i] / 2 + 0.5
        plt.imshow(np.transpose(img.numpy(), (1, 2, 0)))
        plt.title(f"Predicted: {'Dog' if preds[i]==1 else 'Cat'}")
        plt.axis('off')
        plt.show()
    break
```

Predicted: Cat

Predicted: Dog

Predicted: Cat

The model correctly classified common dog/cat breeds. It misclassified blurry or occluded animals. For example, a Persian cat may be misclassified as a dog due to its round face and dark color.

Impact of Model & Data Choices

Factor	**Effect**
Larger model	May improve accuracy but slower training.
More aggressive augmentation	Helps generalize, especially on small datasets.
Fine-tuning vs. frozen features	Fine-tuning gives better accuracy at the cost of more training time.

CIFAR-10 Classification

Extended binary classifier (Dogs vs Cats) to the **CIFAR-10** dataset, a well-known benchmark for multi-class image classification. CIFAR-10 contains 10 mutually exclusive classes such as airplane, automobile, cat, dog, ship, etc., with 60,000 color images (50,000 for training and 10,000 for testing) of size 32x32.

Approach

Transfer learning with a **pretrained ResNet18** model (originally trained on ImageNet) and adapted it for CIFAR-10:

Step	Detail
Architecture	ResNet18 pretrained on ImageNet
Modification	Replaced final layer: nn.Linear(512, 10) for 10 classes
Input Size	Resized CIFAR-10 images from 32x32 to 128x128
Augmentation	Basic transform with resizing and normalization (ImageNet mean/std)
Loss Function	CrossEntropyLoss (with optional class weighting)
Optimizer	Adam, learning rate: 0.001
Training Epochs	5 epochs
Hardware	GPU (CUDA-enabled)

Results

Epoch	Train Accuracy	Train Loss
1	74.49%	1182.08
2	77.44%	1019.02
3	78.69%	971.22
4	78.47%	980.88
5	79.25%	949.21

```
Epoch 1/5 - Loss: 1182.082 - Acc: 74.49%
Epoch 2/5 - Loss: 1019.019 - Acc: 77.44%
Epoch 3/5 - Loss: 971.223 - Acc: 78.69%
Epoch 4/5 - Loss: 980.879 - Acc: 78.47%
Epoch 5/5 - Loss: 949.206 - Acc: 79.25%
```

Final Test Accuracy: 80.40%

Predictions

```python
classes = train_data.classes

dataiter = iter(test_loader)
images, labels = next(dataiter)
outputs = model(images.to(device))
_, preds = torch.max(outputs, 1)

# Show 5 predictions
for i in range(5):
    img = images[i] / 2 + 0.5
    npimg = img.numpy()
    plt.imshow(np.transpose(npimg, (1, 2, 0)))
    plt.title(f"Predicted: {classes[preds[i]]}")
    plt.axis('off')
    plt.show()
```

- Correct classifications (cat, airplane, ship)

- A few borderline mistakes where similar classes were confused (e.g., cat vs dog, truck vs automobile)

Improvements Made

- Resizing for ResNet compatibility
- Added test set evaluation
- Visual validation of model predictions

Data Unbalancing in CIFAR-10

Although CIFAR-10 is originally balanced, we simulated **class imbalance** to reflect real-world conditions. Some classes were artificially underrepresented in sampled mini batches.

Solutions

WeightedRandomSampler: Used a WeightedRandomSampler to rebalance classes during mini-batch creation. Classes with fewer samples were given **higher sampling weights**, ensuring that underrepresented classes are shown more frequently to the model during training. Training batches had a more balanced class distribution, helping the model avoid bias toward majority classes.

Class-Weighted Loss Function: Used CrossEntropyLoss with class weights inversely proportional to class frequency. This ensured that the loss function penalized misclassifications of rare classes more heavily. The model prioritized learning from underrepresented classes without overfitting to them.

References

He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep residual learning for image recognition*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770–778. https://doi.org/10.1109/CVPR.2016.90

Simonyan, K., & Zisserman, A. (2015). *Very deep convolutional networks for large-scale image recognition*. International Conference on Learning Representations (ICLR). https://arxiv.org/abs/1409.1556

Krizhevsky, A. (2009). *Learning multiple layers of features from tiny images* [Technical report]. University of Toronto. http://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

Tan, M., & Le, Q. V. (2019). *EfficientNet: Rethinking model scaling for convolutional neural networks*. In Proceedings of the 36th International Conference on Machine Learning (ICML), 6105–6114. https://arxiv.org/abs/1905.11946

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). *Attention is all you need*. In Advances in Neural Information Processing Systems (NeurIPS), 30. https://arxiv.org/abs/1706.03762

Snoek, J., Larochelle, H., & Adams, R. P. (2012). *Practical Bayesian optimization of machine learning algorithms*. In Advances in Neural Information Processing Systems (NeurIPS), 25. https://proceedings.neurips.cc/paper_files/paper/2012/file/05311655a15b75fab86956663e1819cd-Paper.pdf

PyTorch. (2024). Torchvision models documentation. https://pytorch.org/vision/stable/models.html

Krizhevsky, A. (n.d.). *The CIFAR-10 and CIFAR-100 datasets*. https://www.cs.toronto.edu/~kriz/cifar.html