COMP9444 Project Summary

<Fashion Item Classification using Deep Learning>

<Chengduo Di (z5398380)> <Sipeng Wang(z5499235)> <Po-Hsun Chang (z5408868)> <Jinqi Li(z5405141)>
<Xinyang Chen (z5435860)>

I. Introduction

Automated classification of fashion items is crucial for various applications in the fashion industry. However, accurately classifying fashion items can be challenging due to the wide variety of styles, designs, and visual similarities between different categories. Traditional clothing image style recognition methods mainly rely on successfully extracting effective features, which not only consume a lot of time and effort in processing images, but also have low recognition accuracy. As a fundamental task in the field of computer vision, image classification is aim to identify and classify image content into predefined categories or labels. It is equally true that image classification is the key point of many computer vision applications, such a product categorization, recommendations, and search optimization. Our task aims at developing a deep learning-based image classification model using CNNs to accurately classify fashion items into four different categories and finding their sub categories from the main categories, which improve the efficiency of related work.

II. Related Work

Due to the limited annotations of existing datasets, it is difficult to face the challenges in practical applications. According to He et al. (2016), "The proposed deep residual network (ResNet) architecture enables training of much deeper networks than was previously possible, achieving state-of-the-art performance in image classification tasks while maintaining computational efficiency." This highlights the reliability and utility of the ResNet model in various computer vision applications. However, it also shows that although the ResNet model solves the degradation problem of deep network training through residual blocks, it is still challenging to train as the network depth increases.

Another choice is VGG-16, according to Tammina (2019), "The VGG-16 model, with its deep convolutional architecture and pre-training on large-scale datasets, has proven to be highly effective and reliable for a wide range of image classification tasks, making it a popular choice for transfer learning applications."

III. Methods

We utilized the classic ResNet-34 model, a 34-layer Convolutional Neural Network (CNN) known for its robust image classification capabilities. ResNet-34's architecture allows it to effectively capture hierarchical features from images, making it a suitable choice for our tasks.

In addition to ResNet-34, we also implemented a VGG-16 model for comparison. VGG-16 is another well-known CNN model, characterized by its simplicity and depth, which can capture fine-grained features from images. Due to the better performance of Resnet34, the focus of the report is to introduce the work of resnet34.

IV. Experimental Setup

1) Multi-Task Learning Method(base on Resnet34)

As for Resnet34, We implemented a multi-task learning (MTL) approach to simultaneously handle two related tasks, TaskA and TaskB, leveraging their strong correlation to improve overall model performance. In MTL, we share the majority of the model's layers between tasks while dedicating specific layers to each task to capture task-specific features.

2) Shared Layers

For both tasks, we used the ResNet-34 model up to the final fully connected layer as the shared layers. This allows both tasks to benefit from the rich feature representations learned by the shared layers.

3) Task-Specific Layers

TaskA Specific Layers:

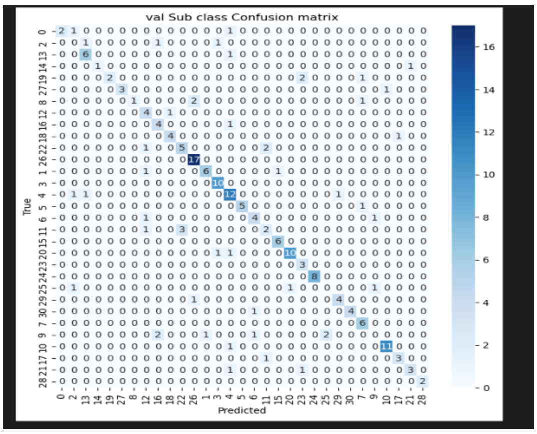
a) Flatten Layer: Converts convolutional output to 1D tensor.
b) Fully Connected Layer 1: nn.Linear(num_ftrs, 128), reduces features.
c) Batch Normalization: nn.BatchNorm1d(128), normalizes output.

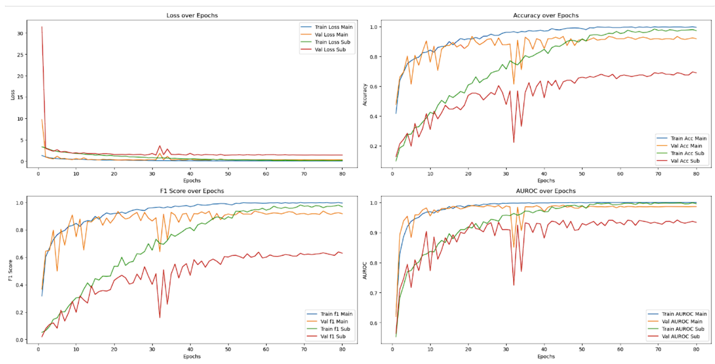
d) ReLU Activation: nn.ReLU(), introduces non-linearity.
e) Fully Connected Layer 2: nn.Linear(128, num_main_classes), final output.
TaskB Specific Layers:
a) Flatten Layer: Converts convolutional output to 1D tensor.
b) Fully Connected Layer 1: nn.Linear(num_ftrs, 256), increases features.
c) Batch Normalization 1: nn.BatchNorm1d(256), normalizes output.
d) ReLU Activation 1: nn.ReLU(), introduces non-linearity.
e) Dropout Layer 1: nn.Dropout(p=0.3), prevents overfitting.
f) Fully Connected Layer 2: nn.Linear(256, 128), reduces features.
g) Batch Normalization 2: nn.BatchNorm1d(128), normalizes output.
h) ReLU Activation 2: nn.ReLU(), introduces more non-linearity.
i) Dropout Layer 2: nn.Dropout(p=0.3), further prevents overfitting.
j) Fully Connected Layer 3: nn.Linear(128, num_sub_classes), final output.
4) Loss Function
We computed the combined loss for both tasks: loss=loss_main+loss_sub This approach ensures that both tasks contribute to the overall model training, promoting a balanced learning process.
We used the Stochastic Gradient Descent (SGD) optimizer with the following parameters:
- Learning rate (lr): 0.01
- Momentum: 0.9
- Weight decay: 0.0005

This optimizer updates model weights based on these parameters.
We also implemented a Cosine Annealing Learning Rate Scheduler:
- Scheduler: optim.lr_scheduler.CosineAnnealingLR(optimizer, T_max=num_epochs, eta_min=0)
- T_max: Number of epochs
- eta_min: Minimum learning rate

This scheduler gradually reduces the learning rate, helping the model converge more effectively.
By using these methods, we aim to leverage shared representations and task-specific features to improve performance on TaskA and TaskB.
5) Direct classification(Base on Vgg16)
This method is a relatively direct classification method and uses different models, but other parts except the basic model are modified similar to the previous method, and the setting of the Scheduler is also different. The learning rate of each epoch is reduced exponentially, and the learning rate of each epoch is 0.9 times of the previous epoch.

V. Results





ResNet34 Multi-Task Learning Results analysis:

Loss over Epochs

- Training Loss (Main): Decreases steadily, indicating effective learning.
- Validation Loss (Main): Follows a similar decreasing trend, suggesting good generalization.
- Training Loss (Sub): Decreases over time, though starts higher and reduces slower.
- Validation Loss (Sub): Exhibits fluctuations but shows an overall decreasing trend.

Accuracy over Epochs

- Training Accuracy (Main): Rapidly increases and stabilizes near 1.0, showing good fit.
- Validation Accuracy (Main): Mirrors training accuracy with minimal overfitting.
- Training Accuracy (Sub): Improves steadily but stabilizes at a lower value.
- Validation Accuracy (Sub): Follows training accuracy trend with performance variability.

F1 Score over Epochs

- Training F1 Score (Main): Quickly increases, indicating balanced precision and recall.
- Validation F1 Score (Main): Closely tracks training F1 score, showing good performance.
- Training F1 Score (Sub): Steadily increases, achieving moderate levels.
- Validation F1 Score (Sub): Improves over time but with more fluctuation.

AUROC over Epochs

- Training AUROC (Main): Increases rapidly, indicating strong class distinction.
- Validation AUROC (Main): Closely follows training AUROC, showing good generalization.
- Training AUROC (Sub): Steady improvement, reaching satisfactory levels.
- Validation AUROC (Sub): Improves with variability, reflecting task complexity.

Confusion Matrix (Validation Sub-Task)

- Diagonal Elements: High correct classification rates for most classes.
- Off-Diagonal Elements: Sparse errors, with some class confusion (e.g., class 2 with classes 1, 4, and 8).

VGG16 Direct Learning Results (Last Two Images)

Class-Specific Performance

- High Performance Classes: Outwear, Trousers, Earrings show high F1 Scores and AUROC.
- Low Performance Classes: Brooch, Back, Neckwear have low F1 Scores and AUROC.
- Intermediate Performance: Most classes fall in between, with moderate performance.

Confusion Matrix (Validation Set)

- Diagonal Elements: Most classes have high correct classification.
- Off-Diagonal Elements: Indicates misclassifications, but they are relatively sparse.

Summary

- ResNet34: Multi-task learning shows effective convergence and generalization with some variability in sub-task performance.

- VGG16: Direct learning shows high performance in specific classes, with overall moderate classification accuracy. Due to space reasons, the results of vgg16 are not analyzed in detail. In general, the performance of vgg16 is worse. The results of vgg16 are shown at the end of the report.

VI. Conclusions

Key Strengths:

1. ResNet34 Multi-Task Learning: Effective for the main task, showing strong learning and generalization.
2. VGG16 Class-Specific Performance: High accuracy for certain classes like Outwear and Earrings.

Key Weaknesses

1. Sub-Task Variability: ResNet34 shows fluctuations in sub-task performance, indicating potential overfitting.
2. Class Confusion in VGG16: Lower performance in classes like Brooch and Neckwear, leading to misclassifications.

Key Limitations:

1. High Computational Cost: ResNet34 requires significant resources.
2. Dataset Constraints: Limited dataset diversity may affect model robustness.

Recommendations for Future Work:

1. Data Augmentation: Improve robustness with more sophisticated techniques.
2. Hybrid Models: Combine ResNet34 and VGG16 strengths.
3. Hyperparameter Optimization: Fine-tune for better performance.
4. Dataset Expansion: Increase size and diversity for better evaluation.

Note: Your project summary should be in at most 4 pages (excluding references). There is no limit on the number of references. Any main text beyond the 4-page limit will be ignored by the evaluator(s).

Reference:

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 770-778.

Tammina, S. (2019). Transfer learning using VGG-16 with deep convolutional neural network for classifying images. International Journal of Scientific and Research Publications (IJSRP), 9(10), 143-150.

Result of vgg16: