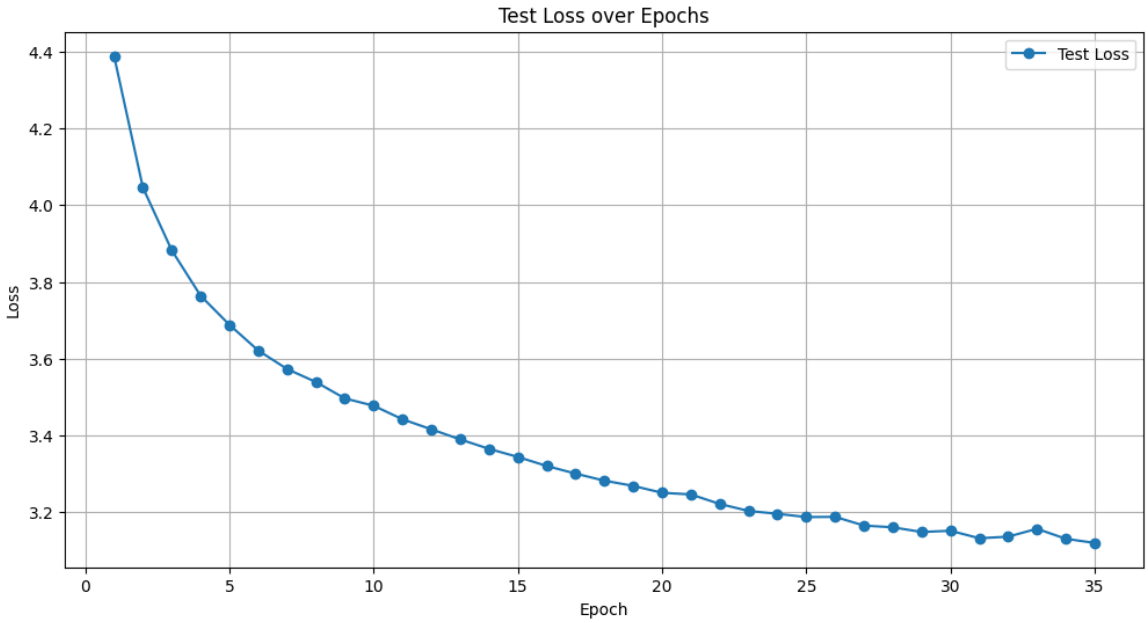
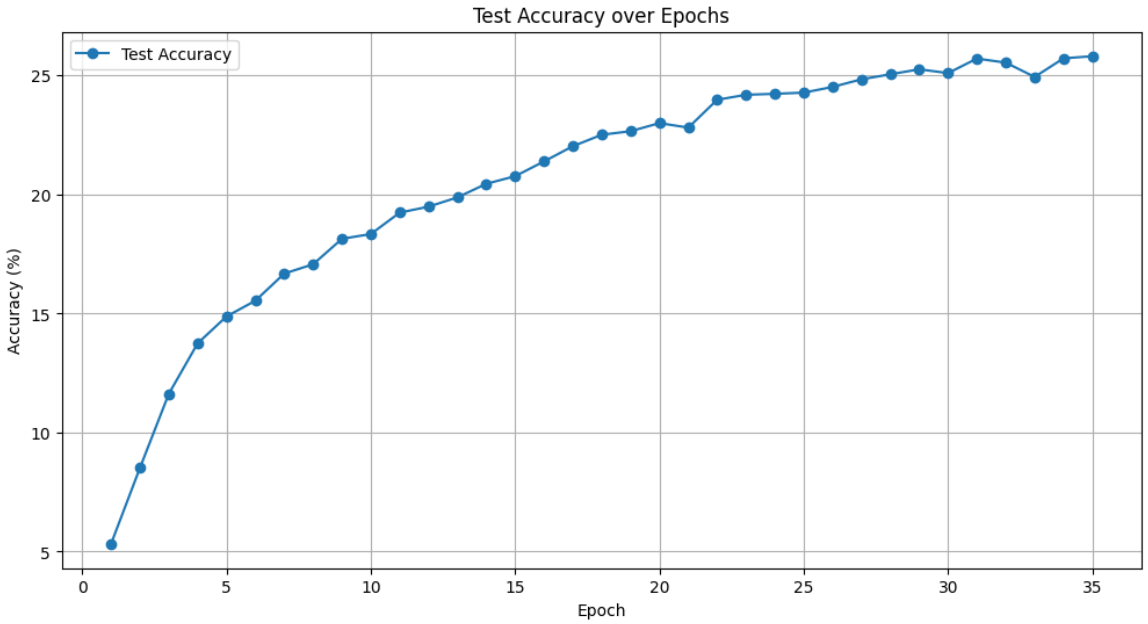
1. DNN



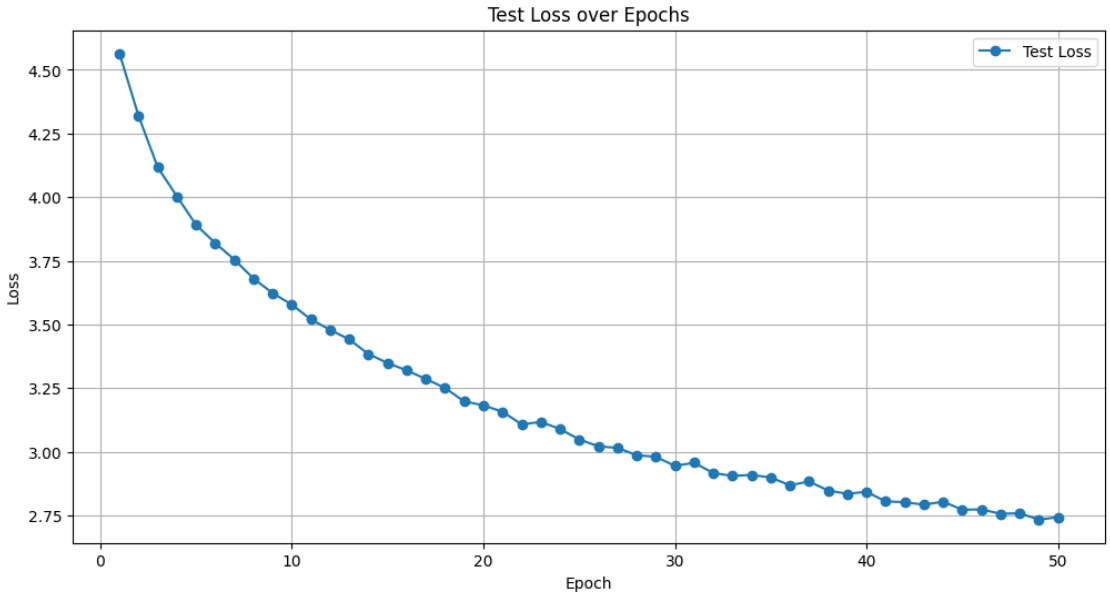


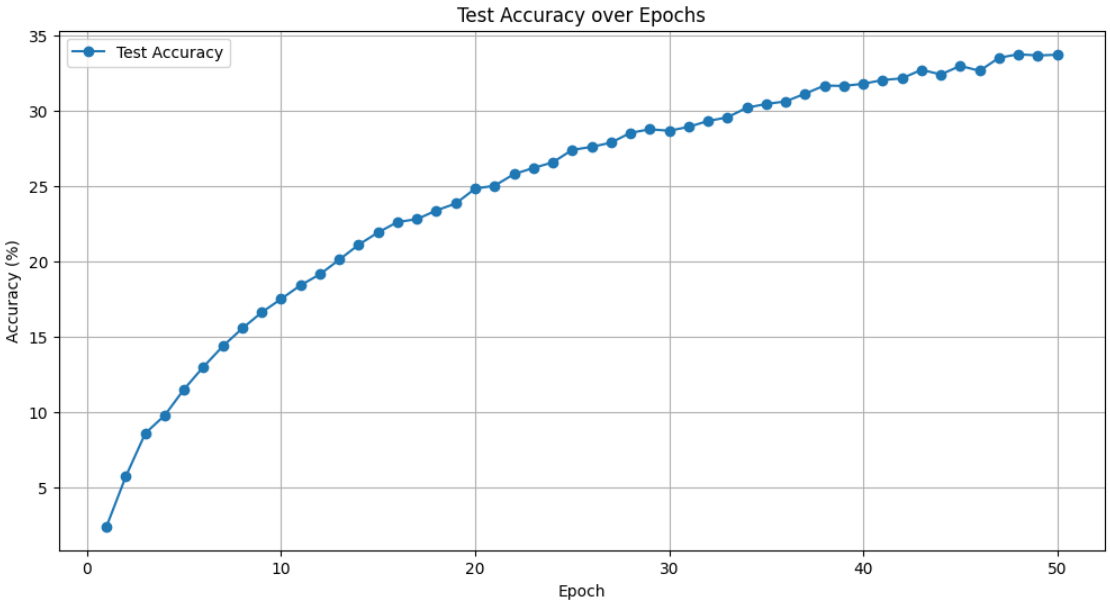
DNN, CNN, ResNet 중 가장 빨리 실행이 완료되었습니다. DNN 모델의 에폭을 100으로 주었더니 정확도는 20대 언저리에서 더 이상 나아지지 않았고 loss값은 점점 낮아지다가 일정 수준이 지나자, U자 형태의 그래프를 그렸습니다.(오버피팅이 발생해서 epoch수를 50으로도 해본 다음 35로 고정하였습니다.

[35] Test Loss: 3.1293, Accuracy: 26.21%

정확도 값이 아주 낮은 양상을 보입니다. DNN의 경우 이미지 위치정보를 고려하지 않아서 학습 이미지가 조금이라도 변동되면 다른 이미지로 인식할 가능성이 있어서 낮은 정확도를 보일 거라고 생각합니다. 또한 이미지 데이터셋의 많은 파라미터가 오버 피팅을 유발할 수 있고 고차원 데이터셋으로 인한 차원의 저주가 생기기 쉬울 것으로 보입니다.

2. CNN





CNN의 경우, 손실값이 DNN에 비해서 더 완만하게 떨어졌고 정확도의 경우도 더 완만하게 올라갔습니다.

[50] Test Loss: 2.7594, Accuracy: 33.91%

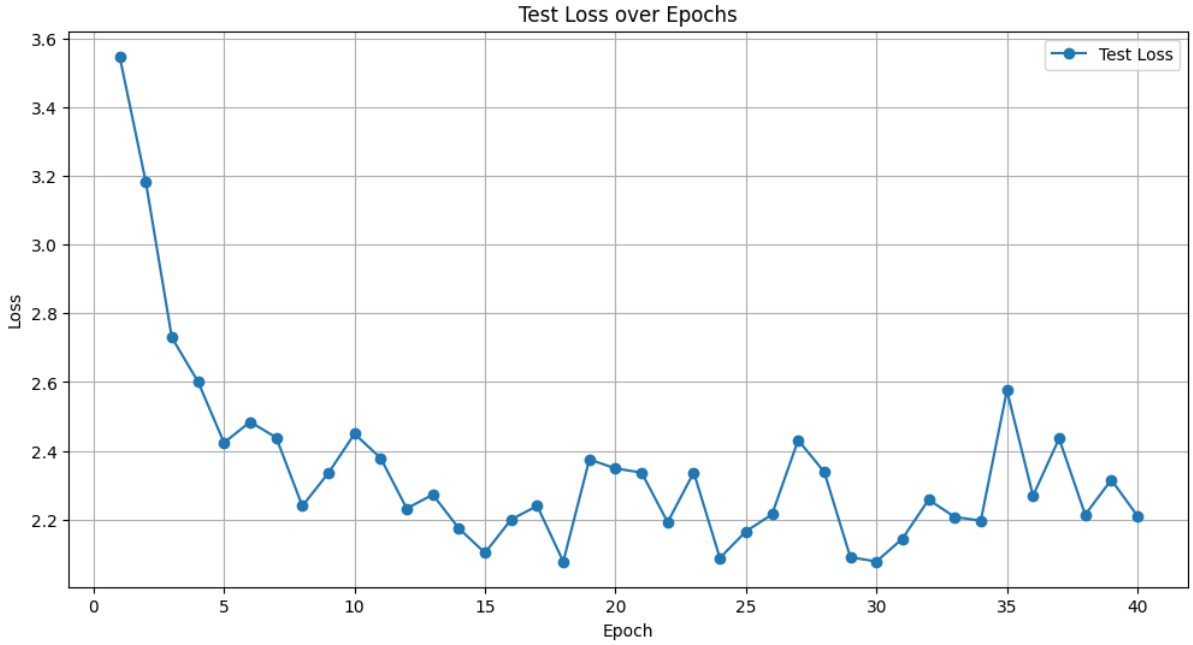
마지막 에폭에서도 CNN 에 비해서는 손실값은 더 떨어지고 정확도는 더 올라가서 성능이 개선됨을 확인할 수 있었습니다. CNN의 경우 Epoch을 50으로 주긴 하였지만

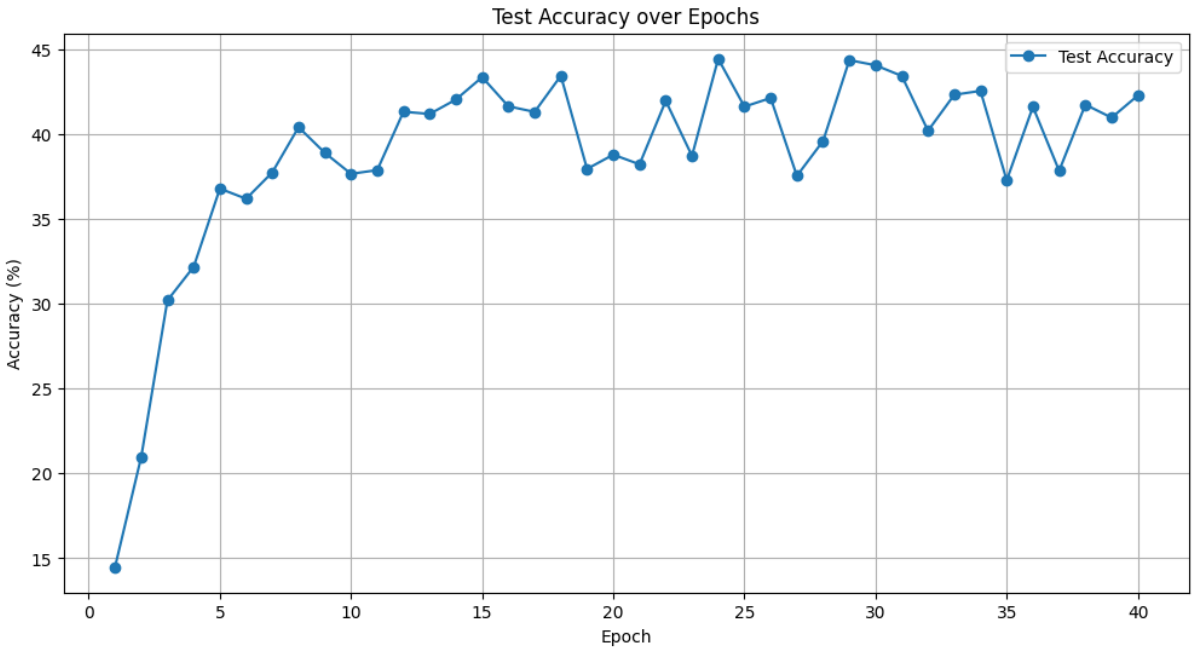
[35] Test Loss: 2.8683, Accuracy: 30.32%

DNN의 최종 에폭 값과 비교해봐도 CNN모델의 성능이 더 이 데이터셋과 적합함을 알 수 있었습니다. 이는 CNN이 DNN과 달리 weight sharing을 하기 때문에 DNN보다 파라미터가 적어지기 때문에 더 좋은 성능을 보이는 것 같습니다.

이 결과로 이미지 데이터셋에는 DNN모델보다는 CNN모델이 더 적합하다고 판단 하였습니다.

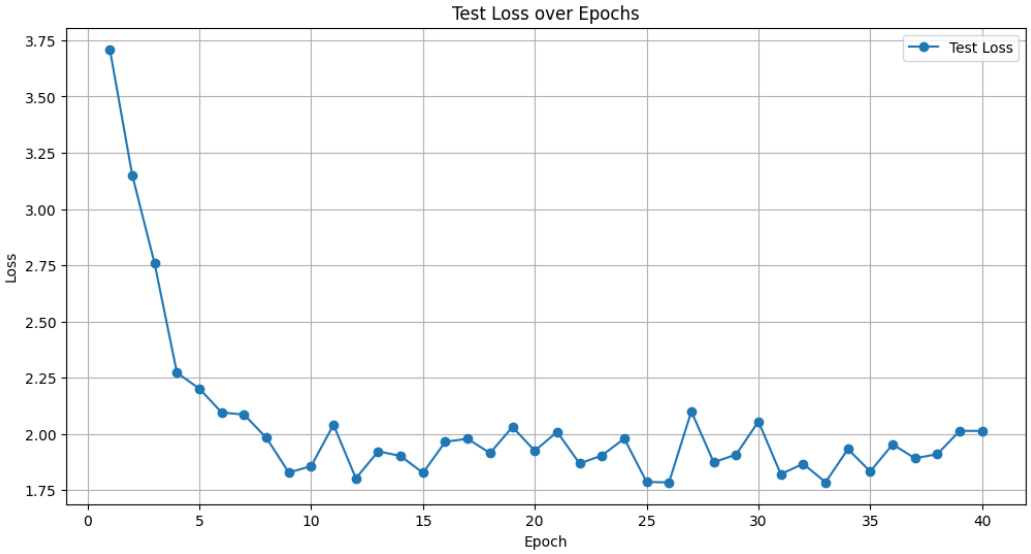
3. ResNet

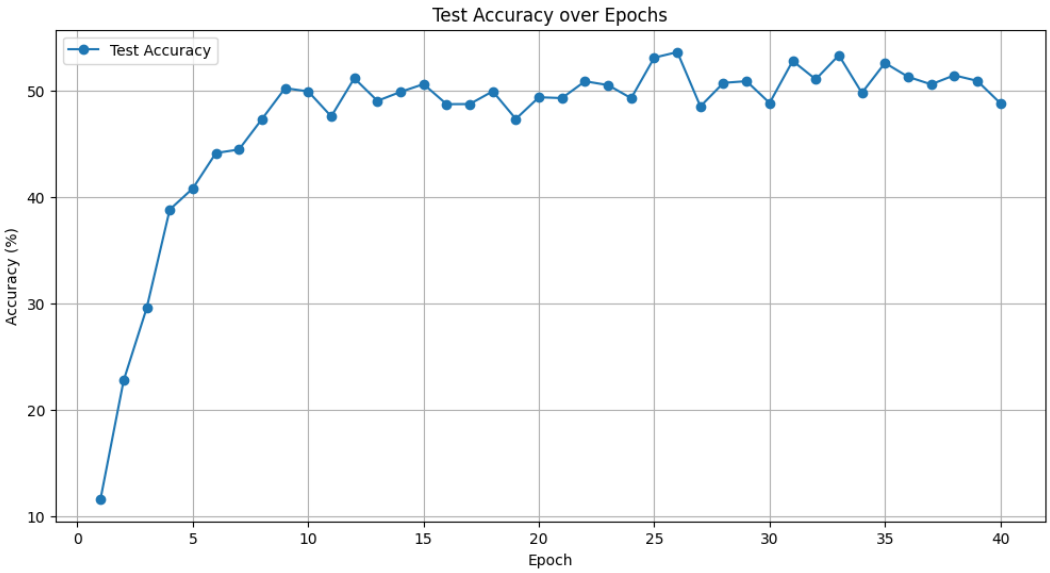




ResNet 의 경우 손실이 급격하게 감소하고 정확도는 급격하게 증가하는 양상을 보였습니다. 12 번째 부터 성능이 40 대로 들어온 다음 가끔씩 30 후반대로 떨어지는 걸 반복하면서 최고 성능은 [24] Test Loss: 2.0881, Accuracy: 44.44%이었고 마지막 때 [40] Test Loss: 2.2108, Accuracy: 42.28%을 기록하였습니다. DNN 이나 CNN 과 달리 그래프의 모양이 가파르고 불안정적인 모양을 보입니다.
하지만 성능 자체는 두 모델보다 더 뛰어난 성능을 보입니다.

4. Custom Model





self.layer1 = self._make_layer(16, 2, stride=1)
self.layer2 = self._make_layer(32, 2, stride=1)
self.layer3 = self._make_layer(64, 2, stride=2)
self.layer4 = self._make_layer(128, 2, stride=2)
기존 ResNet 모델에서 레이어를 추가하고 다운 샘플링을 조절하였습니다.
기존 ResNet에서 테스트할 때 학습률이 0.1보다 값이 커지면 성능이 떨어지고 값이 작아지면 성능 좋아져서 0.1에서 0.05로 값을 바꾸었습니다.
그 결과 기존 모델보다 그래프가 안정화되었고 성능도 26 번째에 [26] Test Loss: 1.7849, Accuracy: 53.58%로 최고 성능을 기록하였고 마지막 시행에서는 [40] Test Loss: 2.0135, Accuracy: 48.78%를 기록하며 기존 ResNet 보다는 좋은 성능을 보였습니다.
Custom 모델이 에폭이 40 번임에도 불구하고 너무 많은 시간을 필요로 하여 더 큰 에폭으로 테스트를 진행하지 못했습니다.