기계학습원론 최종 프로젝트 결과 보고서

2022315965 문준원

1. 서론

1.1 문제와 그 중요성

다중 클래스 분류문제는 머신러닝에서 가장 기본적이면서도 중요한 문제 중 하나로 여겨집니다. 일반적으로 Logistic Regression과 같은 모델을 활용해 다중 클래스를 분류하고, Softmax 함수를 통해 각 클래스의 확률을 계산하여 결과를 나타냅니다. 그러나 클래스 간 유사도가 높은 경우, 즉, Softmax 출력 확률이 각 클래스에 균등하게 분포하거나 예측 결과가 불확실하게 나타날 가능성이 큽니다. 예를 들어, 5개의 유사한 클래스를 분류할 때, 모든 클래스가 동일하게 20%의 확률로 출력된다면, 이는 직관적이지 않을 뿐만 아니라 결과를 신뢰하기 어렵게 만듭니다.

1.2 기존 알고리즘의 한계

현재 사용되는 Cross-Entropy Loss와 기본 Softmax 함수는 각 클래스 간의 독립성을 가정합니다. 이러한 접근법은 클래스 간 유사도나 불확실성을 고려하지 않기 때문에, 유사 클래스 간의 상호작용을 효과적으로 해결하지 못합니다. 결과적으로, 모델이 유사한 클래스를 구분하지 못하고, 불확실한 예측 결과를 제공하여 실질적인 분류 성능을 저하시킬 수 있습니다.

1.3 프로젝트 목표

본 프로젝트는 클래스 간 유사성이 높은 다중 클래스 분류 문제에서 기존 알고리즘의 한계를 극복하고, 예측 결과의 직관성과 신뢰성을 개선하는 것을 목표로 합니다. 이를 위해 다음 두 가지 변형을 제안합니다
첫 번째로, **Logistic Regression 손실 함수 변형**입니다. 클래스 간 유사도를 반영한 새로운 손실 함수를 설계하여, 유사한 클래스 간 혼동을 줄이고 모델 학습의 효율성을 높일 것입니다.
두 번째로, **Uncertainty-Aware Softmax**라는 새로운 함수를 제안합니다. Softmax 함수에 불확실성을 반영하여, 모델의 신뢰도를 명시적으로 전달하고, 직관적인 확률 분포를 제공할 것입니다.

1.4 예상 결과

본 프로젝트를 통해 예상되는 결과는 다음과 같습니다.
첫 번째로, 클래스 간 유사성 문제 해결입니다. 새로운 손실 함수를 통해, 모

델이 유사한 클래스 간의 관계를 학습하여 더 높은 분류 성능을 발휘할 것입니다.
두 번째로, 직관적인 확률 분포 제공입니다. Uncertainty-Aware Softmax를 활용하여 예측 결과의 신뢰성을 향상시키고, 사용자가 결과를 더 직관적으로 이해할 수 있도록 할 것입니다.
세 번째로, 일반화 가능성입니다. 프로젝트에서 제안된 변형 방법은 클래스 간 유사함을 가지는 다중 클래스 분류가 필요한 다양한 데이터셋과 문제에 적용 가능하며, 머신러닝 응용 분야에서의 활용 가능성을 넓힐 것입니다.

1.5 실현 가능성

Logistic Regression과 Softmax 함수는 머신러닝에서 가장 널리 사용되는 알고리즘으로, 변형의 이론적 근거가 명확하고 구현이 비교적 간단하다고 생각합니다. 또한, 실험적으로 검증할 수 있는 적절한 데이터셋(예: Stanford Dogs Dataset)을 사용하여, 제안된 방법의 효과를 실험할 계획입니다. 이를 통해 실질적인 응용 가능성을 확보할 것입니다.

2. 방법

2.1.0 Logistic Regression 변형 : 클래스 간 유사성을 반영한 손실 함수

2.1.1 기존 Cross-Entropy Loss의 한계

1. 기존 Cross-Entropy Loss는 클래스 간의 독립성을 가정하며, 클래스 간 유사성이나 관계를 전혀 반영하지 않습니다.
2. 유사 클래스(예: "푸들"과 "비숑") 간 혼동과 비유사 클래스(예: "푸들"과 "셰퍼드") 간 혼동을 동일하게 처리합니다.

2.1.2 손실 함수 변형

클래스 간 유사도를 반영하기 위해 유사도 행렬(Similarity Matrix)를 손실 함수에 통합합니다.
새로운 손실 함수는 다음과 같이 정의됩니다:

$$L = -\sum_{i=1}^{C}\sum_{j=1}^{C} S_{i,j} \cdot y_i \cdot \log(\hat{y_j})$$

- $S_{i,j}$: 클래스 i와 j간의 유사도를 나타내는 행렬(1에 가까울수록 유사, 0에 가까울수록 다름).
- y_i : 실제 데이터가 속한 정답 클래스에 대한 표시 값.

- $\hat{y}_j$: Softmax를 통해 계산된 클래스 j의 예측 확률.

2.1.3 설계 의도

1. 클래스 간 유사도가 높은 경우(예: "푸들"과 "비숑"), 잘못된 예측에 대해 페널티를 낮춰 모델이 유사 클래스 간의 관계를 학습하도록 유도합니다.
2. 반대로, 유사도가 낮은 경우(예: "푸들"과 "셰퍼드")에는 더 큰 페널티를 부여하여 비유사 클래스 간 혼동을 방지합니다.
3. 유사도 행렬을 설계하게 된 배경은, 수업 시간에 배운 엔트로피(Entropy)의 개념에서 착안하였습니다. 엔트로피는 데이터가 잘 분리될수록 낮아지고, 불순할수록 높아지는 특성을 가집니다. 이러한 개념을 활용해, 클래스 간 유사성을 정량화한 행렬을 설계하고 이를 손실 함수에 통합한다면, 클래스 간의 불확실성을 반영할 수 있을 것이라고 생각했습니다. 특히, 클래스 간 유사도가 높은 분류 문제(예: '푸들'과 '말티즈')에서 모델이 유사한 클래스 간의 혼동을 효과적으로 줄이고, 더 나은 성능을 낼 수 있을 것으로 기대했습니다.

2.1.4 적합성

이 접근법은 클래스 간 유사성이 높은 데이터셋(예: Stanford Dogs)에서 특히 효과적이며, 다중 클래스 분류 문제의 일반적인 한계를 극복할 수 있습니다.

2.2.0 Uncertainty-Aware Softmax

2.2.1 기존 한계

Softmax 함수는 각 클래스의 확률을 독립적으로 계산하며, 예측 불확실성을 명시적으로 표현하지 못합니다.
결과적으로, 예측이 불확실한 경우에도 확률 분포가 균등하게 출력되어, 결과 해석이 어려워질 수 있습니다.

2.2.2 Softmax 변형

Softmax 계산에 **불확실성(Uncertainty)**값을 통합하여, 예측 결과에 대한 신뢰도를 반영합니다.
새로운 Softmax 함수는 다음과 같이 정의됩니다:

$$Softmax(z_i) = \frac{\exp((z_i - u_i)/\tau)}{\sum_{i=1}^{C}\exp((z_j - u_j)/\tau)}$$

- z_i : 클래스 i에 대한 모델 출력(로짓).
- u_i : 클래스 i의 불확실성을 나타내는 값(예: 분산 또는 데이터 복잡성).
- τ : 온도 파라미터로, 확률 분포를 조정.

2.2.3 설계 의도

불확실성이 높은 클래스의 확률을 낮춰, 모델의 신뢰성을 반영하고 예측 결과를 더 직관적으로 만듭니다.
확률 분포가 데이터의 특성을 더 잘 반영하도록 조정됩니다.

2.2.4 적합성

이 접근법은 클래스 간 혼동을 줄이고, 예측 결과의 신뢰성을 개선하는 데 효과적입니다.
예: "푸들"과 "비숑" 간의 불확실성을 낮게 출력하고, 더 명확한 클래스("셰퍼드")에는 높은 확률을 부여합니다.

3. 실험 계획

3.1 실험 목적

본 실험은 제안된 Logistic Regression 변형(손실 함수)과 **Uncertainty-Aware Softmax**가 기존 방식에 비해 다중 클래스 분류 문제에서 더 나은 성능과 직관적인 결과를 제공하는지 검증하는 데 목적이 있습니다. 특히, 클래스 간 유사도가 높은 데이터셋에서 이 접근법이 유사 클래스 간 혼동 문제를 효과적으로 해결하는지 평가할 것입니다.

3.2 데이터셋

Stanford Dogs Dataset의 일부분 사용:

- Number of categories: 7
- Number of images: 1299

```
# 사용할 클래스 리스트 (7개 클래스만)
breed_list = [
    "n02113799-standard_poodle",
    "n02113712-miniature_poodle",
    "n02113624-toy_poodle",
    "n02085936-Maltese_dog",
    "n02086240-Shih-Tzu",
    "n02112018-Pomeranian",
    "n02086079-Pekinese",
```

1. 7개의 클래스(개 품종)로 구성된 이미지 분류 데이터셋.

2. 클래스 간 시각적 유사성이 높아, Softmax의 확률 분포 및 손실 함수 개선 효과를 테스트하기에 적합.

3. 이미지 전처리:
    - 크기 조정: 224×224.
    - 정규화: 픽셀 값을 0~1 범위로 스케일링.
    - 데이터 증강: 랜덤 회전, 랜덤 이동, 좌우 플립.

4. 데이터 분할:
    - 80% 훈련 데이터
    - 20% 테스트 데이터

3.3 비교 실험

Baseline 모델 : Logistic Regression + 기존 Cross-Entropy Loss + 기본 Softmax
- 기존 알고리즘을 사용하여 성능 및 예측 결과를 비교.

제안한 모델 : Logistic Regression + 변형된 손실 함수 + Uncertainty-Aware Softmax
- 클래스 간 유사도를 반영한 손실 함수와 불확실성을 포함한 Softmax를 통합.
- 유사 클래스 간 혼동 문제 해결과 직관적인 확률 분포 제공 여부를 테스트.

3.4 실험 환경

프레임워크:
- Python, TensorFlow/Keras 또는 PyTorch.

하드웨어:
- Google Colab Pro L4 GPU를 활용하여 학습 가속화.

하이퍼파라미터 설정:

- Learning Rate: 0.001
- Batch Size: 32
- Epochs: 20
- Temperature : 1.0 (초기값, 실험적으로 최적화 예정).
- Similarity Matrix: 코사인 유사도를 사용해 클래스 간 관계를 정의.

3.5 성능 평가 지표

Accuracy : 전체 분류 정확도.

F1-Score : 클래스 간 균형 잡힌 평가를 제공.

Calibration Error : 모델 예측 확률과 실제 정답 간의 신뢰성 차이를 측정.

Cross-Entropy Loss : 최종 손실 값 비교.

Confusion Matrix:
- 클래스 간 혼동 정도를 시각적으로 확인.
- 유사 클래스 간 혼동 감소 여부를 분석.

확률 분포 시각화 : Softmax 출력 확률의 직관성과 신뢰성을 평가.

3.6 실험 과정

데이터 전처리:
- 이미지를 224×224크기로 조정하고, 정규화 및 데이터 증강을 수행합니다.

```python
# 이미지 전처리 함수: 크기 조정 및 정규화
def preprocess_image(image_path, target_size=(224, 224)):
    img = cv2.imread(image_path)
    if img is None:
        raise ValueError(f"Image not found: {image_path}")
    img = cv2.resize(img, target_size)  # 크기 조정
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)  # RGB 변환
    img = img / 255.0  # 정규화
    return img
```

```python
# 데이터 증강 정의
train_datagen = ImageDataGenerator(
    rescale=1.0/255.0,
    rotation_range=30,
    width_shift_range=0.2,
    height_shift_range=0.2,
    horizontal_flip=True,
)
```

Baseline 모델 학습:
- 기존 Cross-Entropy Loss와 Softmax를 사용하여 Logistic Regression 모델 학습.
- 성능 지표와 예측 확률 분포 기록.

```
# 모델 정의
baseline_model = Sequential([
    Flatten(input_shape=[224, 224, 3]),  # 입력 크기 224x224x3
    Dense(7),
    Softmax()
])

# 모델 컴파일
baseline_model.compile(
    optimizer=Adam(learning_rate=0.001),
    loss=SparseCategoricalCrossentropy(),
    metrics=[SparseCategoricalAccuracy()]
)

# 모델 요약
baseline_model.summary()
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/layers/reshaping/flatten.py:37: UserWarning:
  super().__init__(**kwargs)
Model: "sequential_16"
```

Layer (type)	Output Shape	Param #
flatten_16 (Flatten)	(None, 150528)	0
dense_16 (Dense)	(None, 7)	1,053,703
softmax_16 (Softmax)	(None, 7)	0

```
Total params: 1,053,703 (4.02 MB)
Trainable params: 1,053,703 (4.02 MB)
Non-trainable params: 0 (0.00 B)
```

제안된 모델 학습:

- 변형된 손실 함수와 Uncertainty-Aware Softmax를 적용하여 학습.
- 동일한 데이터셋, 하이퍼파라미터, 실험 환경 사용.

```
def similarity_ce_loss(y_true, y_pred):
    """
    y_true: (batch,)  정수 라벨 (예: [0, 3, 1, ...])
    y_pred: (batch, C) softmax 결과 (0~1 확률)
    similarity_matrix_tf: (C, C)  (상위 스코프에서 정의)

    수식:
      L = - (1/B) * Σ_{sample=1 to B} [ Σ_{j=1 to C} ( S[i, j] * log(y_pred_j) ) ]
      단, i는 정답 클래스, y_true에서 one-hot으로 얻어짐
    """
    # 1) 정답을 One-hot 변환: (batch, C)
    #    y_true[i]에 해당하는 위치만 1, 나머지는 0
    y_true_one_hot = tf.one_hot(tf.cast(y_true, tf.int32), depth=C)

    # 2) 정답 클래스 i에 해당하는 row: (batch, C)
    #    y_true_one_hot과 S(C,C)를 행렬곱하면, 각 샘플별로 "S[i, :]"를 뽑아온 것과 동일
    sim_row = tf.matmul(y_true_one_hot, similarity_matrix_tf)  # shape=(batch, C)

    # 3) log(y_pred): (batch, C)
    log_pred = tf.math.log(y_pred + 1e-9)

    # 4) element-wise 곱: sim_row * log_pred → (batch, C)
    #    그리고 각 샘플마다 클래스 차원(C)을 sum
    loss_per_sample = -tf.reduce_sum(sim_row * log_pred, axis=1)  # (batch,)

    # 5) 전체 배치 평균
    loss = tf.reduce_mean(loss_per_sample)
    return loss
```

```
# 클래스 개수 C
C = 7

# 1) 유사도 행렬을 임의로 생성/혹은 직접 정의
similarity_matrix = np.random.uniform(0, 1, (C, C))
# 대칭화(선택사항, 문제 정의에 따라 달라질 수 있음)
similarity_matrix = (similarity_matrix + similarity_matrix.T) / 2.0
# 자기 자신과의 유사도 = 1
np.fill_diagonal(similarity_matrix, 1.0)

# 2) 텐서 형태로 변환 (학습 중에는 고정 상수로 사용)
similarity_matrix_tf = tf.constant(similarity_matrix, dtype=tf.float32)
```

성능 비교:

- Baseline과 제안된 모델 간의 성능 지표(Accuracy, F1-Score 등) 및 예측 확률 분포 비교.
- 특히 유사 클래스 간 혼동 정도를 집중 분석.

하이퍼파라미터 튜닝:

- Temperature 값과 Similarity Matrix 가중치 등을 조정하여 최적의 성능을 도출.

3.7 예상 결과

1. 성능 개선 : 제안된 모델이 기존 모델 대비 Accuracy와 F1-Score에서 더 높은 성능을 기록.

2. 직관적 확률 분포 : Uncertainty-Aware Softmax를 통해 유사 클래스 간 확률 분포가 더 직관적으로 표현.

3. 유사 클래스 혼동 감소 : Confusion Matrix에서 유사 클래스 간 혼동이 감소한 것을 확인.

4. 자유 논의

4.1 예상되는 어려움과 해결 방안

1. 유사도 행렬(Similarity Matrix) 설계:

- **어려움**: 클래스 간 유사도를 정량적으로 계산하기 위해 유사도 행렬을 설계하는 과정에서, 적절한 기준(예: 코사인 유사도, 유클리드 거리 등)을 선택하는 데 어려움이 있을 것으로 예상됩니다.

- **해결 방안**:
  - 처음에는 간단한 코사인 유사도를 기반으로 유사도 행렬을 생성하고, 결과에 따라 다른 방법을 실험적으로 적용할 것입니다.

2. Uncertainty-Aware Softmax 파라미터 튜닝:

- **어려움**: 불확실성을 반영하기 위한 u_i 값과 온도 파라미터 τ 의 적절한 설정은 실험적으로 결정해야 하며, 이에 따라 성능이 크게 달라질 수 있을 가능성이 있습니다.

- **해결 방안**:
  - 초반의 실험에서는 단순한 값(예:1)부터 시작해 결과를 점진적으로 개선.

3. 클래스 불균형 문제:

- **어려움**: 데이터셋 내 클래스 간 데이터 수가 불균형할 경우, 특정 클래스의 예측 성능이 저하될 가능성이 높습니다. 예를 들어, 흰색 포메라니안의 학습 이미지가 부족한 상황에서, 모델이 흰색 포메라니안을 정확히 예측하지 못하고 흰색이며 복슬복슬한 다른 강아지(예: 미니 비숑)를 예측할 수 있습니다. 반면, 갈색 포메라니안의 학습 데이터가 충분한 경우에는 98% 이상의 높은 확률로 올바르게 포메라니안을 예측할 가능성이 큽니다. 이는 특정 클래스(흰색 포메라니안)의 데이터 수가 부족해, 모델이 해당 클래스를 충분히 학습하지 못했기 때문입니다.

- **해결 방안**:
  - 데이터 증강 기법을 활용해 소수 클래스의 데이터를 증가해 볼 것입니다.
  - 클래스 가중치를 손실 함수에 반영하여 학습 균형을 맞춤.

4.2 확장 가능성

1. 다양한 데이터셋 적용:

- 특정 데이터셋(예: Stanford Dogs)에 초점을 맞추고 있지만, 클래스 간 유사도가 높은 다른 데이터셋(예: CIFAR-100, Fashion MNIST 등)에도 쉽게 확장 가능합니다.
- 이미지 분류뿐만 아니라 텍스트 분류나 추천 시스템과 같은 다른 도메인에서도 적용 가능성을 탐색할 수 있을 것입니다.

2. 다양한 손실 함수와 Softmax 변형 실험:

- 제안된 손실 함수와 Softmax 변형 외에도, Contrastive Loss 기반의 손실 함수와 결합하여 성능 향상 가능성을 실험할 수 있을 것입니다.
- Softmax와 같은 전체 정규화 방식 외에도, Sigmoid처럼 각 클래스의 독립적인 확률 계산 방식을 사용하여 클래스 간 관계를 분석할 수 있습니다.

3. 모델 확장:

- Logistic Regression 모델 외에 Decision Tree, Support Vector Machines(SVM)과 같은 다른 머신러닝 알고리즘에 손실 함수와 Softmax 변형을 적용해 평가해 볼 수 있습니다.

4.3 실제 적용 시 고려해야 할 사항

1. 계산 비용 : 변형된 손실 함수와 Softmax 계산은 기존 방법보다 약간의 추가 계산 비용을 요구할 수 있습니다. 실제 응용에서는 이러한 계산 비용과 성능 간의 균형을 고려해야 할 것입니다.

2. 예측 결과의 해석 가능성 : Uncertainty-Aware Softmax의 확률 분포는 더 직관적일 수 있지만, 사용자에게 불확실성을 명확히 전달할 수 있는 인터페이스 설계가 필요할 것 같습니다.

3. 모델 일반화 : 제안된 접근법이 특정 데이터셋에서 효과적이라고 해도, 일반화된 문제에 적용하려면 다양한 데이터셋에서의 테스트가 추가적으로 필요하다고 생각 합니다.

4.4 제안 방식의 한계와 보완 방안

1. 한계 : 클래스 간 유사성과 불확실성을 반영하지만, 이미지 분류에 특화된 사전 학습된 CNN 기반 모델에 비해 성능이 낮을 가능성이 있습니다.

2. 보완 방안 : 단일 모델 대신 클래스 간 계층 구조를 반영한 결합 모델(예: Decision Tree와 Softmax의 결합)을 설계하여 성능을 보완할 수 있습니다.

5. 실험 결과

5.0 제안서에서의 변경사항:

1. 데이터셋 변경:

- 초기 계획에서는 Stanford Dogs Dataset의 전체 120개 클래스를 사용하려 했으나, 기계 학습 모델로 이를 분류하는 것은 어려웠습니다.
- 이로 인해 정확도가 3~4%로 매우 낮게 나타났습니다.
- 따라서, 120개 품종 중 외형이 비슷한 7개의 품종을 선별하여 데이터셋을 재구성하였습니다.

2. 성능 평가 지표 변경:

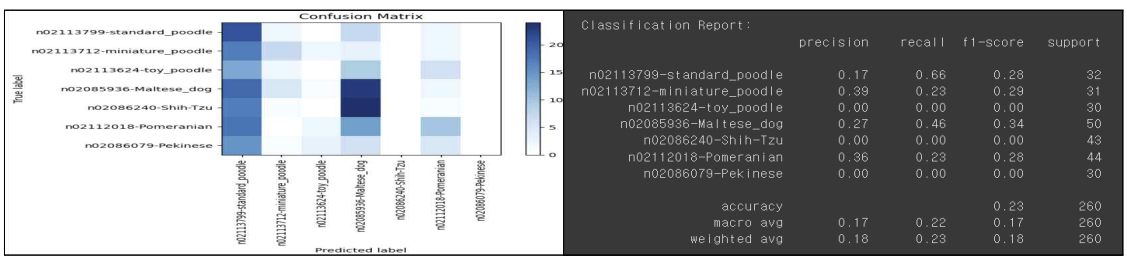
- 초기 계획에서는 Top-k Accuracy를 포함하여 성능 평가를 진행하려 했으나, 제안된 모델이 다룰 클래스 수가 줄어들면서 Top-k Accuracy의 필요성이 낮아졌습니다.
- 따라서 Top-k Accuracy는 최종 성능 평가 지표에서 제외하였습니다.

3. 2.1.3 설계의도 3번 추가

- 아이디어의 배경에 대한 설명을 추가하여 의도를 명확히 이해하시면 좋을것 같아 추가하였습니다.

5.1 Baseline 모델 성능지표와 예측결과:

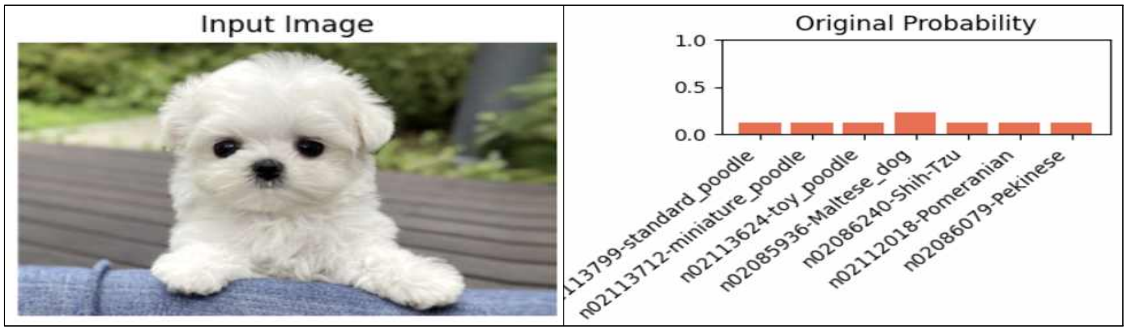
Test Loss: 8.0316
Test Accuracy: 0.2346
F1-Score (weighted) : 0.1801
Calibration Error: 0.1292



Classification Report:

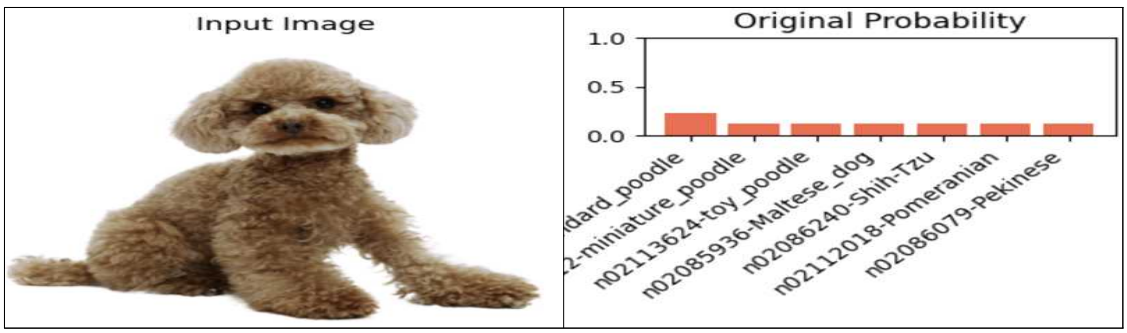
	precision	recall	f1-score	support
n02113799-standard_poodle	0.17	0.66	0.28	32
n02113712-miniature_poodle	0.39	0.23	0.29	31
n02113624-toy_poodle	0.00	0.00	0.00	30
n02085936-Maltese_dog	0.27	0.46	0.34	50
n02086240-Shih-Tzu	0.00	0.00	0.00	43
n02112018-Pomeranian	0.36	0.23	0.28	44
n02086079-Pekinese	0.00	0.00	0.00	30
accuracy			0.23	260
macro avg	0.17	0.22	0.17	260
weighted avg	0.18	0.23	0.18	260

1. 새로운 말티즈 사진에 대한 예측결과



n02113799-standard_poodle : 12.74%
n02113712-miniature_poodle : 12.74%
n02113624-toy_poodle : 12.74%
n02085936-Maltese_dog : 23.55%
n02086240-Shih-Tzu : 12.74%
n02112018-Pomeranian : 12.74%
n02086079-Pekinese : 12.74%

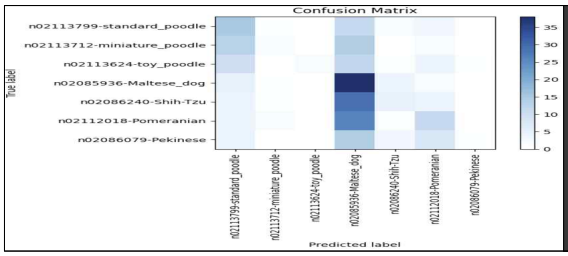
2. 새로운 푸들 사진에 대한 예측결과



n02113799-standard_poodle : 29.51%
n02113712-miniature_poodle : 11.75%
n02113624-toy_poodle : 11.75%
n02085936-Maltese_dog : 11.75%
n02086240-Shih-Tzu : 11.75%
n02112018-Pomeranian : 11.75%
n02086079-Pekinese : 11.75%

5.2 제안된 모델 성능 지표:

Test Loss: 1.8096
Test Accuracy: 0.2846
F1-Score (weighted) : 0.2277
Calibration Error: 0.0415



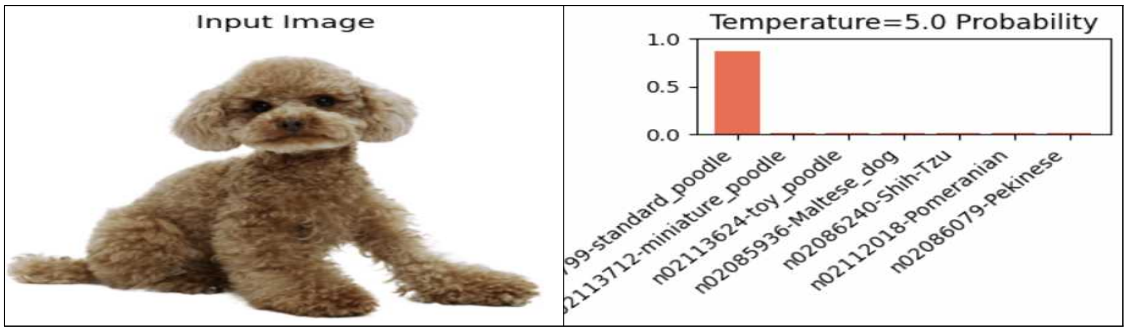
Classification Report:

	precision	recall	f1-score	support
n02113799-standard_poodle	0.27	0.47	0.34	32
n02113712-miniature_poodle	0.29	0.06	0.11	31
n02113624-toy_poodle	1.00	0.07	0.12	30
n02085936-Maltese_dog	0.26	0.76	0.39	50
n02086240-Shih-Tzu	0.31	0.12	0.17	43
n02112018-Pomeranian	0.32	0.25	0.28	44
n02086079-Pekinese	0.50	0.03	0.06	30
accuracy			0.28	260
macro avg	0.42	0.25	0.21	260
weighted avg	0.40	0.28	0.23	260

1. 새로운 말티즈 사진에 대한 예측결과

n02113799-standard_poodle : 2.18%
n02113712-miniature_poodle : 2.18%
n02113624-toy_poodle : 2.18%
n02085936-Maltese_dog : 86.89%
n02086240-Shih-Tzu : 2.18%
n02112018-Pomeranian : 2.18%
n02086079-Pekinese : 2.20%

2. 새로운 푸들 사진에 대한 예측결과



n02113799-standard_poodle : 86.90%
n02113712-miniature_poodle : 2.18%
n02113624-toy_poodle : 2.18%
n02085936-Maltese_dog : 2.18%
n02086240-Shih-Tzu : 2.18%
n02112018-Pomeranian : 2.18%
n02086079-Pekinese : 2.18%

6. 결론:

6.1 Baselin 모델 평가

Baseline 모델의 성능은 전반적으로 낮았으며, 클래스 간 혼동이 큰 문제가 있었습니다. 새로운 이미지에 대한 예측 결과에서도 클래스 간 확률 분포가 고르게 분포되는 경향을 보였습니다.

- **새로운 말티즈 사진 예측:**23.55%의 확률로 "Maltese_dog"을 예측했지만, 나머지 클래스에도 동일한 수준의 확률을 분산시켜 신뢰성이 낮았습니다.
- **새로운 푸들 사진 예측:**"Standard_poodle"로 29.51%의 가장 높은 확률을 예측했지만, 다른 클래스에도 유사한 확률을 분배하였습니다.

이는 Baseline 모델이 클래스 간 유사도와 불확실성을 충분히 고려하지 못해 예측 신뢰도가 낮음을 보여줍니다.

6.2 제안된 모델 평가

제안된 모델은 클래스 간 유사도를 반영한 손실 함수와 Uncertainty-Aware Softmax를 적용하여 학습하였습니다.

제안된 모델은 Baseline 모델에 비해 성능 지표가 전반적으로 개선되었으며, 특히 Calibration Error가 크게 줄어들어 모델의 신뢰성을 향상시켰습니다. 새로운 이미지에 대한 예측 결과에서도 높은 신뢰도를 확인할 수 있었습니다.

- **새로운 말티즈 사진 예측:**86.89%의 확률로 "Maltese_dog"을 정확히 예측하였으며, 다른 클래스에 대한 확률은 매우 낮았습니다.
- **새로운 푸들 사진 예측:**86.90%의 확률로 "Standard_poodle"을 정확히 예측하며, 신뢰도 높은 결과를 보여주었습니다.

6.3 결론 및 향후방향

본 프로젝트를 통해 제안된 모델은 클래스 간 유사도를 반영한 손실 함수와 Uncertainty-Aware Softmax를 통해 기존 Baseline 모델의 한계를 극복하고 성능을 향상시킬 수 있음을 확인했습니다. 특히, Calibration Error가 크게 감소함으로써 모델의 신뢰성이 크게 향상되었으며, 새로운 이미지 예측에서도 높은 정확성과 직관적인 확률 분포를 보여주었습니다.

결론

1. **성능 개선:**
   제안된 모델은 Baseline 모델 대비 Test Accuracy, F1-Score, Calibration Error 등에서 전반적으로 개선된 결과를 보여줬습니다. 특히, 클래스 간 혼동 문제가 완화되고, 불확실성 반영을 통해 예측 결과의 신뢰도를 높이는 데 성공했습니다.
2. **유의미한 결과:**
   새로운 데이터(말티즈 및 푸들 사진)에 대한 예측 결과에서, 제안된 모델은 높은 정확도로 해당 클래스를 예측하며 직관적이고 신뢰할 수 있는 확률 분포를 제공하였습니다.

향후 방향

1. 다양한 데이터셋 적용 및 일반화 가능성 탐색

현재 프로젝트는 Stanford Dogs 데이터셋에 초점을 맞추고 있지만, 제안된 접근법은 다른 데이터셋에도 적용 가능성을 갖습니다. 예를 들어, 클래스 간 유사도가 높은 CIFAR-100이나 Fashion MNIST 같은 데이터셋에서의 확장 가능성을 평가할 수 있습니다. 또한, 이미지 분류뿐만 아니라 텍스트 분류, 추천 시스템 등 다른 도메인에서도 적용 가능성을 탐구할 필요가 있습니다.

2. 다양한 손실 함수와 Softmax 변형 실험
제안된 손실 함수 외에도 Contrastive Loss와 같은 손실 함수와의 결합을 통해 성능 향상을 탐색할 수 있습니다.
Softmax 방식 외에도 Sigmoid 기반 독립 확률 계산 방식을 사용하여 클래스 간 관계를 보다 정밀하게 분석할 가능성을 실험할 수 있습니다.

3. 모델 확장 및 추가 평가
현재 연구는 Logistic Regression 기반으로 진행되었지만, Decision Tree, SVM, 또는 사전 학습된 CNN 모델에 손실 함수와 Softmax 변형을 적용하여 추가적인 평가를 수행할 필요가 있습니다. 특히, kaggle또는 github의 사전 학습된 CNN 모델을 사용한 사람들과의 정확도 차이가 크다는것을 확인하였습니다.

4. 계산 비용 및 해석 가능성
- 제안된 접근법은 기존 손실 함수 및 Softmax 방식에 비해 약간의 계산 비용이 증가할 수 있습니다. 이를 실제 환경에서 적용할 때 계산 비용과 성능 간의 균형을 고려해야 합니다.
- Uncertainty-Aware Softmax가 제공하는 불확실성 정보를 사용자에게 효과적으로 전달할 수 있는 인터페이스 설계도 중요한 과제입니다.

5. 한계와 보완 방안
- 제안된 접근법은 클래스 간 유사성과 불확실성을 반영하지만, 사전 학습된 CNN 모델과 같은 복잡한 모델에 비해 성능이 낮을 가능성이 있습니다. 이를 보완하기 위해 클래스 간 계층 구조를 반영한 결합 모델(예: Decision Tree와 Softmax의 결합) 또는 하이브리드 모델을 설계해볼 수 있습니다.

본 프로젝트를 통해 다중 클래스 분류 문제에서 클래스 간 유사성과 불확실성을 반영한 손실 함수와 Softmax 변형의 가능성을 확인하였다고 생각합니다.

7. 출처 및 참고 문헌:

[1] Stanford Dogs Dataset, Kaggle. URL: https://www.kaggle.com/datasets/jessicali9530/stanford-dogs-dataset/data

[2] 소프트맥스(Softmax) 함수,
URL: https://syj9700.tistory.com/38

[3] Softmax Classifier의 이해 & Python으로 구현하기,
URL: https://yamalab.tistory.com/87

[4] 3.5. 이미지 분류 데이터 (Fashion-MNIST),
URL: https://ko.d2l.ai/chapter_deep-learning-basics/fashion-mnist.html

[5] 교수님 강의 교안.

[6] CIFAR-10 and CIFAR-100 datasets,
URL: https://www.cs.toronto.edu/~kriz/cifar.html

[7] Cross-Entropy Loss Function,

URL:
https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e

[8] [Open DMQA Seminar] Calibration of Deep Neural Networks,
URL: https://www.youtube.com/watch?v=GybzFyQBJIA

[9] [논문 리뷰] On Calibration of Modern Neural Network,
URL:
https://velog.io/@hwanee/%EB%85%BC%EB%AC%B8-%EB%A6%AC%EB%B7%B0-On-Calibration-of-Modern-Neural-Network

[10] OpenAI GPT, 대화형 인공지능 모델(내용작성 및 검토에 활용),
URL: https://openai.com/chatgpt