**Introduction**

Content moderation is crucial for maintaining a safe and appropriate environment on social media platforms, especially those aimed at children. This project aims to develop an image classification model capable of identifying inappropriate content categories such as gore, violence, nudity, and substances, while distinguishing them from neutral images. To achieve this, we utilized transfer learning with EfficientNet, a state-of-the-art convolutional neural network architecture.

**Class Selection**

For this project, we defined five classes:

1. **Neutral**: Images that are safe and appropriate for children.
2. **Gore**: Images depicting blood, injury, or severe bodily harm.
3. **Violence**: Images showing acts of physical aggression or combat.
4. **Nudity**: Images containing explicit content or nudity.
5. **Substances**: Images depicting drugs, alcohol, or related paraphernalia.

The selection of these classes was guided by common content moderation policies and the need to cover a broad range of potentially harmful content that children should not be exposed to.

**Data Gathering**

Data gathering involved collecting images for each of the five classes. This process included:

* **Neutral**: Sourced from publicly available image datasets and safe stock photos.
* **Gore**: Obtained from medical imagery databases and content tagged with warnings on various platforms.
* **Violence**: Collected from hate speech, symbols and cultist propaganda.
* **Nudity**: Retrieved from datasets used for adult content detection and filtering.
* **Substances**: Compiled from images of drug use, alcohol consumption, and related materials.

Each class was populated with approx 4000 images to ensure a balanced dataset while neutral . The images were preprocessed and labeled accordingly.

**Data Preparation**

We then split the data into training (80%), validation (10%), and test (10%) sets. Using torchvision.datasets.ImageFolder and torch.utils.data.random\_split, we ensured each subset had the correct proportions.

### Model Selection: EfficientNet

EfficientNet was chosen due to its balance of accuracy and efficiency. It scales the model dimensions (depth, width, resolution) uniformly with a compound coefficient, leading to superior performance with fewer parameters compared to other architectures.

### Transfer Learning with EfficientNet

We experimented with different versions of EfficientNet (B0 to B7). For practical considerations, including computational resources and training time, we chose EfficientNet-B0. This model offered a good trade-off between performance and resource requirements.

#### Implementation Steps:

1. **Data Transformation**: Applied random resizing, horizontal flips, and normalization to augment and standardize the data.
2. **Model Loading**: Loaded the pre-trained EfficientNet-B0 model from efficientnet\_pytorch.
3. **Fine-Tuning**: Replaced the final fully connected layer with a new one matching the number of our classes (5). The new layer was trained while freezing the rest of the model initially, then fine-tuning the entire network.
4. **Training**: Trained the model using the Adam optimizer with a learning rate scheduler to adjust the learning rate dynamically. The model was trained for 20 epochs with a batch size of 8.
5. **Validation**: Evaluated the model's performance on the validation set after each epoch to prevent overfitting and ensure generalization.
6. **Testing**: Finally, the model's accuracy was assessed on the test set to determine its real-world effectiveness.

### Results

The trained model achieved the following accuracies:

**Training Accuracy: 0.9677**

**Validation Accuracy: 0.9737**

**Test Accuracy: 0.9722**