Multi-Label Movie Genre Classification Using Transfer Learning

Burak Arayıt
burak.arayit@ozu.edu.tr

Efe Tolga
efe.tolga@ozu.edu.tr

Spring 2025

Abstract

This project investigates the problem of multi-label classification of movie genres from poster images using transfer learning. We compare several convolutional neural network (CNN) architectures including ResNet50, EfficientNet, and VGG16. Our dataset consists of labeled movie posters downloaded from Kaggle. After preprocessing and one-hot encoding of genre labels, we train each model and evaluate them using multi-label metrics such as F1-score and precision. VGG16 with fine-tuning provided the best performance, demonstrating the effectiveness of deeper pretrained architectures with customized head layers.

1 Introduction

Movie posters are the main attractive element of a movie. The audience first interacts with the movie poster, and if they find it appealing, they usually proceed to explore more—like watching the trailer or reading about the movie. There are underlying visual elements and subtle messages hidden in posters that hint at the movie's tone, style, or genre. We've decided to investigate this visual attractiveness by starting with genre classification based on posters.

Since a movie can belong to multiple genres at once, the task becomes even more complex than standard single-label classification. Instead of relying on old-school feature extraction techniques, we directly use convolutional neural networks (CNNs) to extract features and detect the genre of a movie poster in one go! In this project, we apply transfer learning on famous pretrained models like ResNet50, EfficientNetB0, and VGG16. The main advantage of transfer learning is that it reduces training time significantly and lowers dependence on powerful GPU resources.

Our main goal is to classify movie genres with high precision using visual input only and to compare how different CNN architectures perform on this task. Along the way, we also observe the effects of fine-tuning, hyperparameter changes, and the model's ability to generalize over multi-label genre data. This task also has real-world relevance in applications like recommendation systems, content tagging, and automated metadata generation for large-scale media databases.

2 Related Work

There have been several attempts in the literature to classify movie genres using poster images, each with a different take on how to extract meaningful visual features.

One of the early studies by Ivašić-Kos et al. [4] focused on low-level visual features like color histograms, edge detection, and face counts to classify posters into genres using algorithms like Naïve Bayes and RAKEL. They approached the problem as multi-label but transformed it into a single-label task for easier modeling. Their best performance was around 14% for perfectly matching

both genre labels, and about 67% when at least one genre was correct, showing that even simple features can carry useful information.

Rasheed and Shah [7] tackled the problem from movie trailers instead of posters. They used audio-visual features such as shot length, motion, and lighting to classify movies into genres like action, comedy, or horror. Their method relied heavily on cinematic principles and low-level cues, which worked surprisingly well on a small dataset of movie previews.

Moving toward deep learning, Chu and Guo [2] combined CNN-based visual features with object detection using YOLOv2. They fine-tuned a network that merged these features to handle multi-label classification, achieving around 18.7% accuracy across 23 genres. Their approach showed the benefit of mixing semantic object cues with global visual features.

Kundalia et al. [5] took a different approach by simplifying multi-label classification into a top-3 prediction task. They built a large, balanced dataset and used transfer learning with InceptionV3. Thanks to careful data design and knowledge transfer, their model hit an impressive 84.8% accuracy, although it was technically reframed as a single-label task during training.

Dewidar's CS231n project [3] focused on directly applying ResNet101, VGG19, and AlexNet on poster images. Without any fancy balancing or preprocessing, his best model (ResNet101) reached about 42% accuracy, which is solid considering the complexity of the task and the multi-label nature.

Lastly, Barney and Kaya [1] combined ResNet34 with a custom CNN architecture and even tried ML-KNN as a baseline. Their experiments showed ResNet34 performing best (around 38% accuracy), but also highlighted the limitations caused by class imbalance in the dataset. They emphasized metrics like recall and Hamming loss, which are crucial in multi-label tasks.

All of these works show that poster-based genre classification is definitely feasible, but results depend heavily on how the dataset is prepared, what features are extracted, and how multi-label output is handled. Our work builds on these ideas by comparing different pretrained models using raw poster images, aiming for both solid performance and simple implementation.

3 Data

We used the publicly available *Movie Posters by Genre* dataset from Kaggle [6], which provides a large collection of movie poster images grouped by genre folders. Each image may appear in multiple genre directories, making this a naturally multi-label classification problem.

To preprocess the data, we first created a mapping from each image filename to its associated genres. This allowed us to flatten the folder structure and correctly handle overlapping genre labels. We extracted all unique genres from the dataset and assigned each one a binary index to form multi-hot vectors. Each image was thus represented by a binary vector where each dimension corresponds to a genre, with 1 indicating presence.

We organized this information into a clean CSV file where each row includes the filename and its one-hot encoded genre vector. The final label file `multilabel_posters.csv` serves as the ground truth for model training. We finally obtained total of 33,887 poster images for training.

All images were resized to 224×224 pixels and normalized to $[0, 1]$ range. The dataset was split into training, validation, and test sets with a typical 70:15:15 ratio. While some genres like Drama and Comedy were highly represented, others had relatively fewer examples. To address this imbalance during training, we used metrics like macro-averaged F1-score rather than relying solely on accuracy.

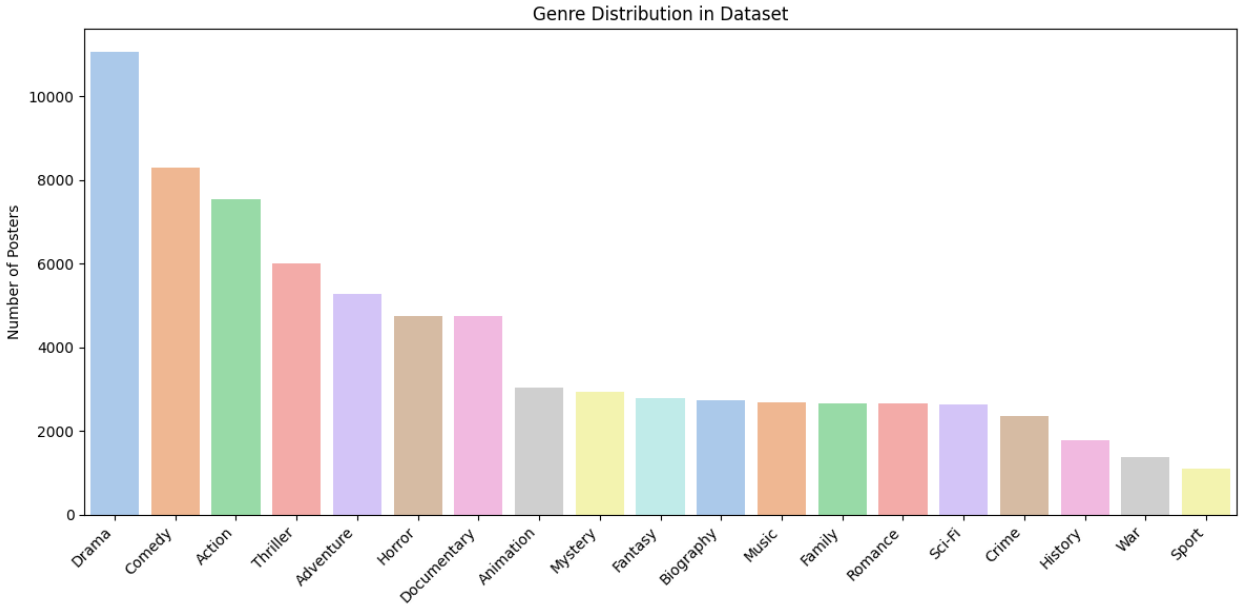

Figure 1: Genre distribution of the dataset

Genre Index	Genre Name
0	Action
1	Adventure
2	Animation
3	Biography
4	Comedy
5	Crime
6	Documentary
7	Drama
8	Family
9	Fantasy
10	History
11	Horror
12	Music
13	Mystery
14	Romance
15	Sci-Fi
16	Sport
17	Thriller
18	War

Table 1: One-hot vector genre encoding used in our dataset. Each index corresponds to a genre; a movie's label is a binary vector of length 19.

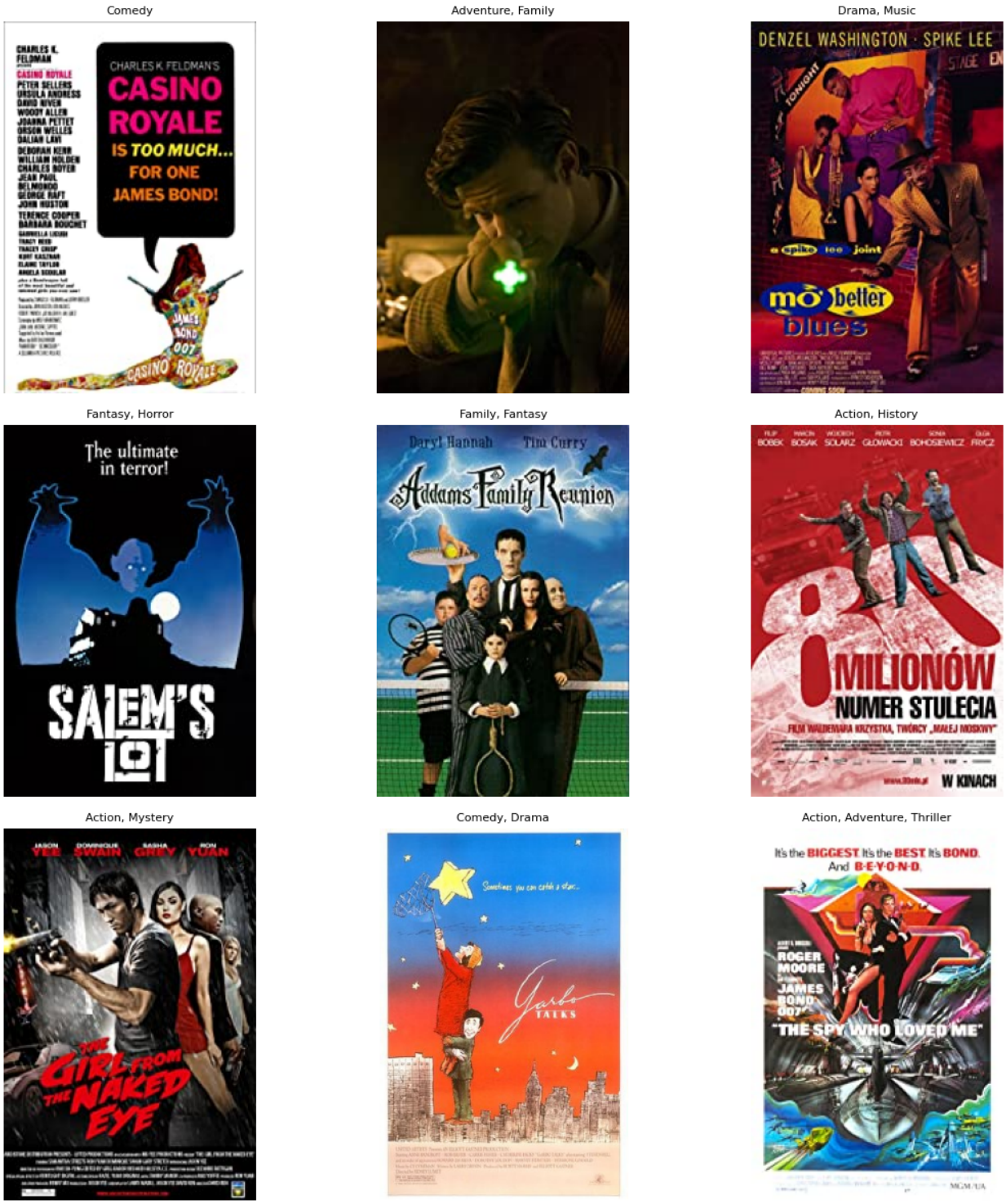

Figure 2: Sample posters from the dataset

4 Methods

We address multi-label movie genre classification from poster images using transfer learning with CNN-based architectures.

4.1 Data Preprocessing

First attempt was to reshape the dataset into well-defined and trainable format. For this purpose all poster images were resized to 224×224 pixels and normalized to the $[0, 1]$ range by dividing RGB pixel values by 255. Since each image can belong to multiple genre; we created a multi-hot encoded label vector, where each dimension represents a genre.

The dataset was split into 70% training, 15% validation, and 15% test sets, stratified by genre distribution. Genre imbalance was mitigated through macro- and micro-averaged metrics, and in some cases via weighted loss functions.

To enhance generalization and reduce overfitting, we applied data augmentation techniques during training. The augmentation pipeline included random horizontal flipping, slight rotations (up to ± 15 degrees), random affine transformations with small translations and scale jittering, and color distortions such as brightness, contrast, saturation, and hue adjustments. These transformations were applied only to the training set, while the validation and test sets used deterministic resizing and normalization. All images were converted to tensors and normalized to the $[-1, 1]$ range using the mean and standard deviation values $[0.5, 0.5, 0.5]$ for each RGB channel.

4.2 Model Architectures

We evaluated four CNN architectures: **ResNet50**, **EfficientNet-B0**, **EfficientNetV2-S**, and **VGG16**. Each model was initialized with pretrained ImageNet weights. The final classification layer of each architecture was removed and replaced with a fully connected layer of 19 outputs and a sigmoid activation function:

$$\sigma(z) = \frac{1}{1 + e^{-z}}, \quad \hat{\mathbf{y}} = \sigma(\mathbf{z}) = [\sigma(z_1), \ldots, \sigma(z_{19})]$$

This configuration allows each output neuron to predict the probability that the input image belongs to the corresponding genre, enabling multi-label classification.

ResNet50: A 50-layer deep residual network consisting of convolutional blocks with identity shortcut connections that help mitigate the vanishing gradient problem. Each residual block learns a function $F(x)$ and adds it to the input x via $F(x)+x$, enabling stable training of deeper networks. ResNet50 contains approximately 25 million parameters.

EfficientNet-B0: A compact and efficient CNN that uses a compound scaling method to uniformly scale depth, width, and resolution. It is composed of inverted residual bottleneck blocks with squeeze-and-excitation optimization. EfficientNet-B0 has approximately 5.3 million parameters, making it efficient for limited-resource environments.

EfficientNetV2-S: A more recent architecture that improves training speed and accuracy through fused-MBConv blocks and progressive learning. It features roughly 21 million parameters and achieves state-of-the-art performance on various benchmarks. We used blocks 4–7 for fine-tuning in our experiments.

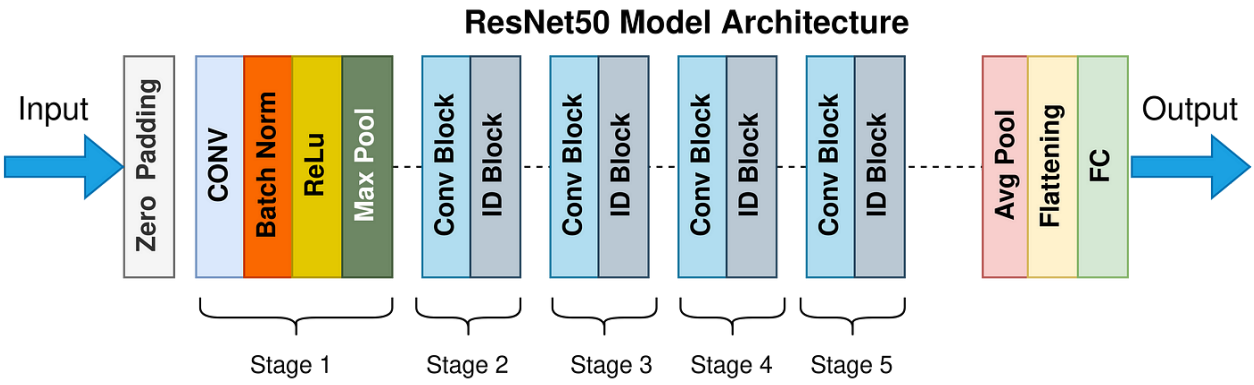

Figure 3: Resnet50 Architecture

VGG16: A classical deep CNN consisting of 13 convolutional layers and 3 fully connected layers, all using 3×3 convolution kernels and ReLU activations. It has approximately 138 million parameters. While deep, it lacks modern improvements like batch normalization and residual connections, making it less efficient.

All models were adapted for our task by replacing the original softmax classifier with a multi-label sigmoid classifier and trained using binary cross-entropy loss.

4.3 Training Strategy

We followed a two-stage transfer learning strategy:

- **Stage 1 (Frozen Backbone):** The feature extractor was frozen and only the classifier head was trained for 10 epochs using Adam optimizer with learning rate $\eta = 10^{-4}$.
- **Stage 2 (Fine-tuning):** Selected blocks (e.g., 4–7 for EfficientNetV2-S) were unfrozen and trained for up to 90 epochs with $\eta = 10^{-5}$. Learning rate decay (StepLR with $\gamma = 0.5$) was applied.

While the default strategy involves staged training, ResNet50 was also evaluated using full fine-tuning as a baseline.

All models were trained using mini-batches of size 32. Early stopping based on validation macro F1-score was employed to prevent overfitting.

4.4 Threshold Optimization

To convert model outputs into binary genre predictions, we applied class-specific thresholding:

$$\tilde{y}_i = \begin{cases} 1 & \text{if } \hat{y}_i \geq t_i \\ 0 & \text{otherwise} \end{cases}$$

Thresholds $t_i \in (0, 1)$ were optimized on the validation set to maximize F1-score for each genre individually. This improves performance on underrepresented genres compared to the default threshold of 0.5.

4.5 Evaluation Metrics

Model performance was evaluated using three main F1-score variants, along with precision and recall:

- **Micro F1-score**: Aggregates contributions of all classes to compute average metrics:

$$\text{Precision}_{\text{micro}} = \frac{\sum_i TP_i}{\sum_i TP_i + FP_i}, \quad \text{Recall}_{\text{micro}} = \frac{\sum_i TP_i}{\sum_i TP_i + FN_i}$$

$$\text{F1}_{\text{micro}} = \frac{2 \cdot \text{Precision}_{\text{micro}} \cdot \text{Recall}_{\text{micro}}}{\text{Precision}_{\text{micro}} + \text{Recall}_{\text{micro}}}$$

- **Macro F1-score**: Computes F1-score independently for each class, then averages:

$$\text{F1}_{\text{macro}} = \frac{1}{C} \sum_{i=1}^{C} \frac{2 \cdot \text{Precision}_i \cdot \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i}$$

- **Sample-based F1-score**: Computes the F1-score for each instance individually and averages over the dataset.

- **Per-class Metrics**: We also report class-specific precision, recall, and F1-score to assess performance across both common and rare genres.

5 Experiments and Results

5.1 ResNet50 Experiments

Considering what we wrote in our proposal and our task at hand, naturally we had a couple of options to choose from when it comes to picking these models. To handle this project, it was decided that the ideal way to do it would be to use transfer-learning, that is obviously obtaining pretrained and created models, and unfreezing certain layers of them for our situation to get our desired results. This is something that was decided considering the depths and complexity of these models and how difficult it would be to create an ideal model from scratch that would outperform these complex models that many have worked on, especially for this complex multi-labeling problem that is present here. It was thought that creating new models and working on tuning them would be an entirely different project altogether so that was skipped for this project. We first established a strong baseline using the pre-trained ResNet50 model. The classifier head was modified for multi-label classification by replacing the final fully connected layer with a sigmoid-activated output layer corresponding to 19 genres. As it was mentioned in the methods section, Resnet50 is one of the older models that is used here, it is not expected to obtain the best possible results that can be obtained with it, which is why its a good baseline model. It is wise to improve on it upon seeing the results for it, but also this should be a pretty decent model, it is not expected to see it give terrible results.

5.1.1 Baseline Model (No Class Balancing)

The process for this was basically trying to find the right epoch number that is ideal for the results. Initial runs were used with small 5-10 epochs to see how it worked. The layers were unfrozen so the model could be trained properly, because before that small F1 scores were achieved since its just a

random pre-trained model at that point not giving us the right weights therefore not solving our problem. The model was trained for 50 epochs using `BCEWithLogitsLoss` without class balancing. It was seen that the best results were around epoch 12, and the rest were more or less excess epochs. Looking at the epoch results, the training loss naturally kept decreasing but the validation loss kept increasing. Although considering that this is a multi-label problem, the validation loss as a metric doesn't quite tell the whole story, which is why the F1 results for each epoch was observed. It was seen that around the 12 epoch, the F1 score peaks and just never improves later on, making it redundant. Which is why it was decided that the best validation performance occurred in one of the runs at epoch 12:

- **Macro F1-score:** 0.4035
- **Micro F1-score:** 0.4621
- **Samples F1-score:** 0.4198

These results show the average performance for these F1 scores, here is a more detailed version for it for each genre using the criteria:
Table 2: Per-class performance of the baseline ResNet50 model (trained for 12 epochs, threshold = 0.5)

Genre	**Precision**	**Recall**	**F1-score**
Action	0.62	0.50	0.55
Adventure	0.57	0.37	0.45
Animation	0.90	0.78	0.83
Biography	0.40	0.32	0.35
Comedy	0.62	0.57	0.59
Crime	0.35	0.27	0.30
Documentary	0.59	0.41	0.49
Drama	0.60	0.49	0.54
Family	0.47	0.31	0.37
Fantasy	0.31	0.25	0.28
History	0.34	0.16	0.21
Horror	0.57	0.53	0.55
Music	0.45	0.26	0.33
Mystery	0.41	0.21	0.28
Romance	0.46	0.33	0.39
Sci-Fi	0.35	0.33	0.34
Sport	0.57	0.31	0.40
Thriller	0.41	0.38	0.39
War	0.38	0.14	0.21
Micro Avg	0.54	0.41	0.47
Macro Avg	0.49	0.36	0.41
Samples Avg	0.45	0.41	0.41

Not necessarily in this run we see here but in the initial runs we did with lower epochs, we saw that the model showed a lot lower scores for rare genres. The issue with that is the dataset available for this case is imbalanced, certain genres have a lot more data than some rare ones like War movies.

Later we will see an improvement on that with the Balanced model, the values here are somewhat better than the previous ones not shown, but seems to leave a lot to be desired. Considering the computational cost, the baseline model seems to be a good model regardless. The metrics used here should be explained above in detail, when we look at them we see that some genre achieve a lot better results than some other, once again due to the imbalanced data present, like looking at War and History we see the lowest values out of all, certainly when compared to some others like Animation genre. The main one is the F1 scores. Some of them like the Animation genre is actually quite good, but the others leave room for improvement. I guess here Animation is actually the best because its pretty distincive or unique, like its easy for the model to recognize a drawn character, as opposed to real life characters which is the case for any other category. The higher Micro results show that this is more of an imbalanced dataset, favoring the common genres, but the model is still somewhat fairly judging the genres and does a good job. If we get into specifics, we can obviously say that Animation is great, Drama is actually disappointing considering that it has a lot of data. Romance seems to be confused with Drama or Comedy a lot, and then some expected bad performances like War for small data. One tuning that was decided for this is the threshold tuning.

Threshold tuning was applied to the baseline ResNet50 model after training for 12 epochs, using the validation set to determine per-class thresholds that maximized F1-score. This was done to improve the results, especially the rare ones, it is explained later. The revised evaluation metrics are shown below:

Genre	**Precision**	**Recall**	**F1-score**
Action	0.57	0.61	0.59
Adventure	0.47	0.47	0.47
Animation	0.92	0.75	0.83
Biography	0.39	0.33	0.36
Comedy	0.62	0.57	0.59
Crime	0.33	0.38	0.35
Documentary	0.51	0.53	0.52
Drama	0.53	0.63	0.58
Family	0.43	0.42	0.42
Fantasy	0.29	0.31	0.30
History	0.23	0.27	0.25
Horror	0.56	0.55	0.55
Music	0.33	0.28	0.31
Mystery	0.34	0.25	0.29
Romance	0.38	0.46	0.42
Sci-Fi	0.43	0.34	0.38
Sport	0.53	0.34	0.41
Thriller	0.36	0.54	0.43
War	0.26	0.20	0.22
Micro Avg	0.48	0.49	0.48
Macro Avg	0.45	0.43	0.44
Weighted Avg	0.48	0.49	0.48
Samples Avg	0.45	0.49	0.45

Table 3: Genre-level evaluation metrics for threshold-tuned baseline ResNet50 model (Epoch 12).

Threshold tuning was applied post-training to improve genre-level performance. The thing with this is that in multi-label classification, each genre or label gets a separate score, probability between 0-1, after applying sigmoid to the model's output logits. Of course, these probabilities need to be binarized to 0 and 1, the most common for this is said to be a fixed threshold, which in default is 0.5, but that doesn't work great with imbalanced datasets. When we have common genres like Drama, the model will be confident and it will have higher output probabilities, similarly for rare ones like War, its not as confident so now it will give lower probabilities. If the threshold is 0.5 for both of these groups, it might under-predict the rare ones or over-predict common ones. At least in principle that was the idea. So we updated our predictions with this thresholding to at least squeeze out a couple more points for these scores at the very least.

The main results obtained from this are:

- **Macro F1-score:** 0.44
- **Micro F1-score:** 0.48
- **Samples F1-score:** 0.45

Compared to the untuned baseline, threshold tuning gave us small but still respectable improvements.. The macro F1-score improved from **0.41** to **0.44**, and micro F1 from **0.47** to **0.48**. These gains indicate better performance on both rare and common genres. Also, genres like *Thriller*, *Drama*, and *Animation* seen some increased recall, while *Romance* and *Documentary* achieved a better balance between precision and recall. We see Action improved quite a bit from 0.55 to 0.59, similarly Drama, Romance, Crime, etc. There are a lot of improvements, there are also only slight improvements to some of the other results here like War, Animation, Music, etc. It seems that this is around where the model is limited to, but the tuning did improve our results quite a bit and showed some good results. Some of them improved due to increased recall, some others due to precision, but either way this is something we use later down the line as well.

5.1.2 Balanced Model (With `pos_weight`)

To mitigate the class imbalance, we computed a posweights vector and retrained ResNet50. So the idea behind this is, with BCEWithLogitsLoss, the posweights parameter adjusts the loss only for positive labels. In a way it basically says that getting a positive label wrong is more costly than getting the negative one wrong, but the dataset is not balanced, so we might want to award the rare ones getting right positive labels; that was the logic behind it. The model can possibly learn to ignore these rare genres, and instead just improve the overall accuracy, because that's the goal of it. But if the weighting on the positives for these rare genres are higher, then it will be too costly for the model, and it will have to not ignore them as much. The way its calculated is given in the code, but in principle it just gives higher posweights for rarer genres. The false negatives for War or Sport is a lot more important than, for example, Drama. Again, a lot of runs were made, then finally this model was trained for 100 epochs and achieved its best performance at epoch 61:

- **Macro F1-score (no tune and tuned):** 0.41 and 0.43
- **Micro F1-score (no tune and tuned):** 0.45 and 0.46
- **Samples F1-score (no tune and tuned):** 0.40 and 0.43

And the criteria for all genres, not tuned and tuned:

Table 4: Balanced ResNet50: Left = No Threshold Tuning, Right = With Threshold Tuning

No Threshold Tuning

Genre	Prec.	Rec.	F1
Action	0.55	0.47	0.51
Adventure	0.44	0.45	0.44
Animation	0.85	0.77	0.81
Biography	0.39	0.28	0.33
Comedy	0.59	0.55	0.57
Crime	0.35	0.36	0.35
Documentary	0.51	0.48	0.50
Drama	0.59	0.48	0.53
Family	0.40	0.38	0.39
Fantasy	0.28	0.24	0.26
History	0.29	0.19	0.23
Horror	0.60	0.40	0.48
Music	0.36	0.31	0.33
Mystery	0.25	0.18	0.21
Romance	0.37	0.42	0.39
Sci-Fi	0.44	0.31	0.36
Sport	0.53	0.39	0.45
Thriller	0.46	0.31	0.37
War	0.36	0.24	0.29
Micro Avg	0.49	0.41	0.45
Macro Avg	0.45	0.38	0.41
Samples Avg	0.43	0.41	0.40

With Threshold Tuning

Genre	Prec.	Rec.	F1
Action	0.48	0.60	0.53
Adventure	0.42	0.47	0.45
Animation	0.88	0.75	0.81
Biography	0.39	0.28	0.33
Comedy	0.52	0.65	0.58
Crime	0.30	0.47	0.37
Documentary	0.48	0.53	0.50
Drama	0.50	0.60	0.55
Family	0.39	0.40	0.39
Fantasy	0.21	0.41	0.28
History	0.20	0.34	0.25
Horror	0.45	0.63	0.53
Music	0.34	0.33	0.34
Mystery	0.23	0.30	0.26
Romance	0.30	0.59	0.40
Sci-Fi	0.37	0.40	0.38
Sport	0.63	0.36	0.46
Thriller	0.37	0.56	0.44
War	0.51	0.20	0.29
Micro Avg	0.42	0.52	0.46
Macro Avg	0.42	0.47	0.43
Samples Avg	0.42	0.51	0.43

So looking at these results, its actually not amazing when we look at it at first, the micro, macro and samples for F1 all either stayed the same or slightly decreased. But when we look at it in detail, we actually do see something nice, whether it is something desired is something that needs to be decided. But we actually see that the rarer genres got a lot better results than the baseline model. We see War one of the rarest genres go from 0.21 to 0.29 F1 score, huge increase in precision and increase in recall. For simplicity sake just comparing the tuned versions of baseline and balanced, Sport gets a lot better, Music gets better, and then some other genres we do see some improvements, but the problem is we also see some of them get worse results, especially genres like Action. The rare genres didn't consistently get better either, Mystery got worse, History stayed the same, etc. It is hard to argue that this worked out great and definitely improved, because we did get some improvement in rare cases, the overall did get slightly worse, and the common genres got a little bit worse. So arguably, it did somewhat do its job, the predictions are more balanced, the rare genres did mostly get better while the common got worse, so I guess the success of both models is a bit subjective. Because something like the War genre did get a lot better, but its still not exactly really accurate. I think considering the computational time for balanced was a lot more, it is fair to prefer the baseline model, even though the rare genres are predicted a lot worse with it. And of course, the tuning gave us better results as it can be seen. The class imbalance issue was somewhat addressed but left a lot of room for improvement.

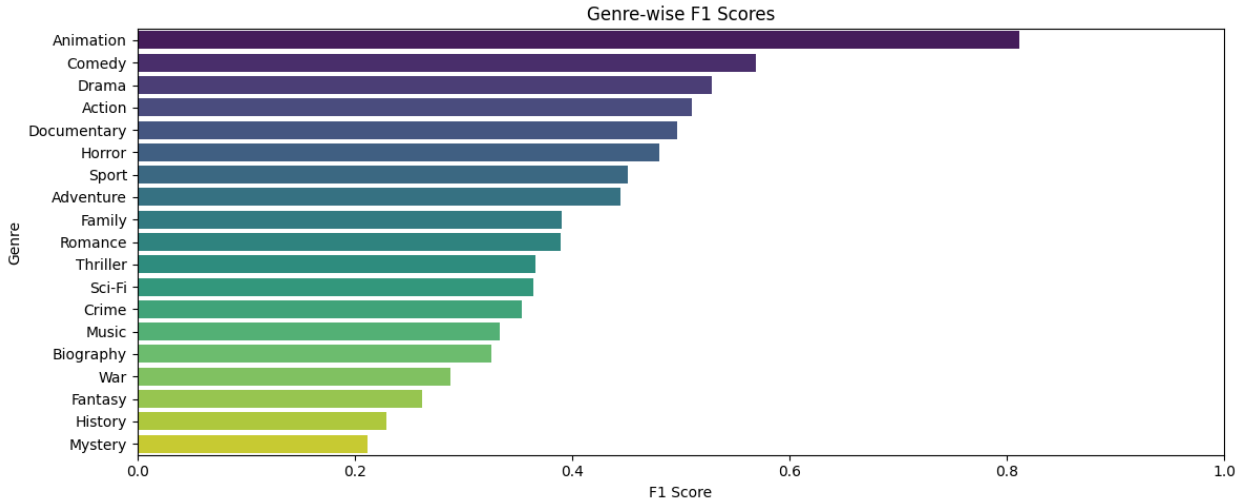

Figure 4: Resnet50 Balanced Model F1 scores for Genres

We can see the scores more clearly here, it seems like Animation is quite a bit more accurate than anything else. The actual results of the model's prediction can be seen here below:

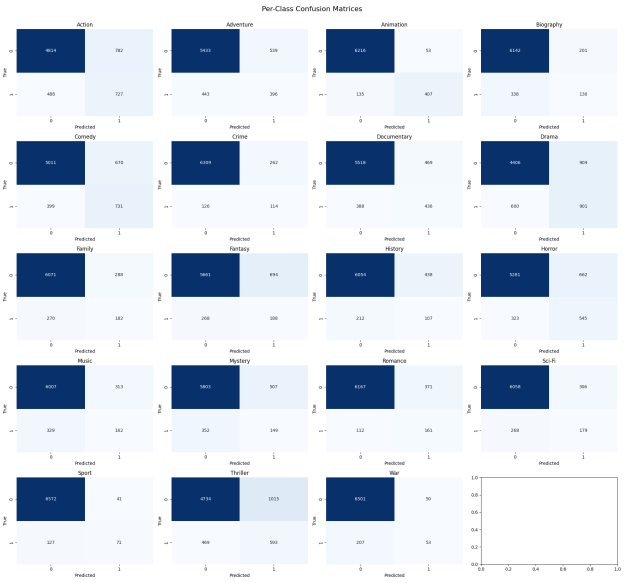

Figure 5: Per-Class Confusion Matrices of Balanced Resnet50

This is a nice way of seeing what the model predicted and what the right answer was, in the notebook there are some other plots like the threshold plots, but another one is this losses and F1 scores plot:

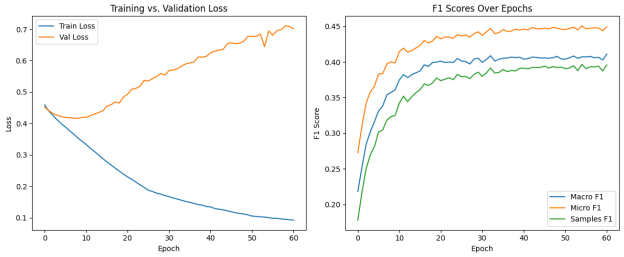

Figure 6: Loss evolution and F1 scores over epochs

This is an interesting plot considering how the validation loss keeps on increasing. From what we have worked on so far, one would think this is a terribly failing model, its seen that even from the very first epochs present, this is just totally overfitting in the most obvious way as the validation loss unlike the training loss increases. The validation loss from our loss function measures how well the raw predicted probabilities match the truth. But the thing done here for this multi-class labeling problem is the raw probabilities are now evaluated, they are binarized with certain thresholds. This means the loss doesn't exactly align with the F1 scores, which just cares about the correctly predicted 1s and 0s. Another aspect is the data imbalance, considering the imbalance, the loss might look great even if it completely misses the rare genres, but the F1 scores actually evaluate this problem a lot better. I suppose the only reason that the loss is useful here is to make sure something unexpected doesn't occur, if the loss explodes to much then there might be a problem for example. But overall we can see the F1 scores here, its clearly stable after a certain epoch. Here is one final evaluation for the model:

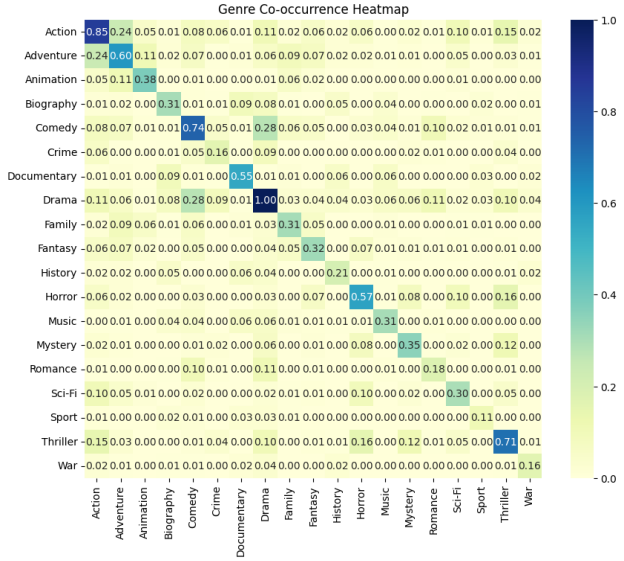

Figure 7: Genre Co-occurence Map

This map shows the co-occurence of the genres, how the model predicted both of them at the same time.

5.2 EfficientNet-B0: Efficient, Smaller and More Modern Model

In addition to deep ResNet50 model, EfficientNet-B0 was used, its more compact and computationally efficient, its a lot newer, faster, smaller model (5.3M parameters). It's not the biggest model, that is seen from the fact that the Resnet50 model had around 25M parameters, but the expectation from this model is a lot more.

Training Strategy. We adopted a two-phase training regime for EfficientNet-B0:

1. **Stage 1 – Frozen Backbone (10 epochs):** All backbone layers were frozen and only the classifier head was trained using Adam optimizer with a learning rate of 1×10^{-4}. The goal was to train the final fully-connected layer to output logits for multi-label genre classification without disturbing the previous layers that are obviously pre-trained.

2. **Stage 2 – Fine-tuning Top Blocks (20 epochs):** We then unfroze the top two blocks (layers 6 and 7) of the EfficientNet-B0 backbone and fine-tuned them alongside the classifier using a reduced learning rate of 1×10^{-5}. This unfreezing helps the model adapt high-level features to the target dataset while preventing the model from forgetting low level filters. This is of course just basic transfer learning, it is not efficient or even accurate to retrain the whole model. The things that make the difference should be at the top blocks since they are higher-level layers anyway like expected, so the base levels should be more than fine. And the other aspect is of course the overfitting, when its all unfrozen, more overfitting can be seen.

In general similar things were done as it was done with the Resnet50 model. But essentially here, the model was set, all the necessary codes were written. Multiple runs were tried with the model, initially just smaller epoch results just to see, later it was decided to try 10 frozen and 20 unfrozen epochs for the model. We will see the other models, more epochs weren't run because the V2-S model gave better results. The final results for EfficientNet-B0 showed:

- **Macro F1 (threshold-tuned):** 0.55
- **Micro F1 (threshold-tuned):** 0.53
- **Samples F1 (threshold-tuned):** 0.52

EfficientNet-B0 with Per-Class Threshold Tuning

The following classification report summarizes the performance of the EfficientNet-B0 model after applying per-class threshold tuning:

Table 5: Classification Report – EfficientNet-B0 (Threshold Tuned)

Genre	Precision	Recall	F1-score
Action	0.58	0.64	0.61
Adventure	0.52	0.55	0.54
Animation	0.90	0.84	0.87
Biography	0.30	0.55	0.39
Comedy	0.65	0.67	0.66
Crime	0.44	0.61	0.51
Documentary	0.54	0.68	0.60
Drama	0.56	0.70	0.62
Family	0.54	0.56	0.55
Fantasy	0.32	0.38	0.35
History	0.33	0.44	0.38
Horror	0.62	0.68	0.65
Music	0.43	0.42	0.42
Mystery	0.35	0.43	0.39
Romance	0.48	0.74	0.58
Sci-Fi	0.47	0.47	0.47
Sport	0.54	0.42	0.47
Thriller	0.44	0.60	0.51
War	0.37	0.49	0.42
Micro Avg	0.51	0.61	0.55
Macro Avg	0.49	0.57	0.53
Weighted Avg	0.52	0.61	0.56
Samples Avg	0.50	0.60	0.52

These are post threshold results, we just directly skipped to that since we understand the effect of the tuning by now. When we look into these results, we can see a massive improvement over the ResNet50 model, it went from 0.40s to 0.50s. WE can actually see every single genre improve their F1 scores, except only Sport seems to be somewhat stable. Other than that, this model simply gave better results, and even better with threshold tuning. Also the F1 results improved in general, especially for rare genres, this is good because it was desired to have a more balanced report, the model did improve but it didn't just boost the common genres, the rare genres like War was around 0.22 for the baseline model, now it is at 0.42. Once again, it seems like more epochs would have improved the results for this model, but given the fact that the V2-S looked better and with the hardware limitations, it was decided to just go on with the V2-S and be fine with only 30 epochs for B0, it is expected to see at least a couple more points higher for this one. Also the training logs might show different numbers, but that's because it used a different evaluation metric, the real numbers are seen in the report that is generated. Similarly the V2-S model will be seen later.

5.3 EfficientNetV2-S: Modern Model-Best Results of the Project

Our final and most successful model was **EfficientNetV2-S**, one of the more modern models used here. It has certain different layers and improvements that the older models didn't have. It is a larger model than the B0, with around 21 million parameters, it is fast, optimized, efficient, and on paper definitely the best model for this purpose from the ones we used.

Training Methodology. We employed a two-stage training strategy designed to gradually increase learning capacity while preserving the pretrained features:

1. **Stage 1 – Frozen Backbone (10 epochs):** All convolutional layers in the `features` module were frozen, and only the final classification head was trained with an Adam optimizer (learning rate: 1×10^{-4}). This is of course the same thing we did with the previous model.
2. **Stage 2 – Progressive Fine-tuning (90 epochs):** We unfreezed the topmost blocks of the network for fine-tuning. Specifically, blocks labeled `4`, `5`, `6`, and `7` in the backbone were set to `requires_grad = True`, while earlier blocks remained frozen to avoid overfitting. The model was trained using a reduced learning rate of 1×10^{-5} with a learning rate scheduler (StepLR with `step_size` $= 15$, $\gamma = 0.5$). This is actually for the final model that was used. Initially, only 6-7 blocks were unfrozen as it will be seen.

The initial step was to run it for low epochs to see the model work. It was run for 20 epochs in total, much more promising results came up which is why it was run again for more and more epochs, some of the runs can be seen in the notebook. The results for the model ran for 50 epochs (10 frozen, 40 unfrozen) with only 6-7 unfrozen came out as:

EfficientNetV2-S (Blocks 6–7 Unfrozen, 50 Epochs)

Genre	Precision	Recall	F1-Score
Action	0.49	0.58	0.53
Adventure	0.37	0.54	0.44
Animation	0.87	0.79	0.83
Biography	0.21	0.47	0.29
Comedy	0.60	0.57	0.59
Crime	0.32	0.47	0.38
Documentary	0.48	0.60	0.53
Drama	0.51	0.61	0.56
Family	0.33	0.52	0.40
Fantasy	0.21	0.43	0.28
History	0.17	0.34	0.23
Horror	0.51	0.62	0.56
Music	0.30	0.38	0.33
Mystery	0.18	0.48	0.27
Romance	0.32	0.63	0.43
Sci-Fi	0.39	0.37	0.38
Sport	0.56	0.44	0.50
Thriller	0.35	0.60	0.45
War	0.27	0.32	0.29
Micro Avg	0.39	0.55	0.46
Macro Avg	0.39	0.51	0.43
Weighted Avg	0.43	0.55	0.47
Samples Avg	0.39	0.55	0.43

Table 6: Per-class metrics for EfficientNetV2-S with 6 and 7 blocks unfrozen (50 epochs).

When these results are observed, it is seen that the results are similar to our Resnet50 Model, which is why it was decided to run with 4-5-6-7 layers unfrozen instead. It seems that the last 2 blocks

weren't necessarily enough to get the best values, and also needed more epochs. Weirdly the log results using a different evaluation method did give us better log results than the B0, hence why it was decided to continue with this model instead. Either way, the next bit with 4-5-6-7 unfrozen and 100 epochs gives us a much better result, that is, the best results obtained yet:

EfficientNetV2-S (Blocks 4–7 Unfrozen, 100 Epochs, Threshold Tuning)

Genre	Precision	Recall	F1-Score
Action	0.72	0.74	0.73
Adventure	0.65	0.72	0.68
Animation	0.97	0.93	0.95
Biography	0.77	0.65	0.70
Comedy	0.78	0.78	0.78
Crime	0.84	0.71	0.77
Documentary	0.81	0.82	0.81
Drama	0.68	0.71	0.70
Family	0.86	0.75	0.81
Fantasy	0.80	0.65	0.71
History	0.82	0.67	0.74
Horror	0.82	0.81	0.81
Music	0.75	0.75	0.75
Mystery	0.68	0.63	0.65
Romance	0.84	0.76	0.80
Sci-Fi	0.85	0.74	0.79
Sport	0.90	0.73	0.81
Thriller	0.55	0.68	0.60
War	0.89	0.71	0.79
Micro Avg	0.74	0.74	0.74
Macro Avg	0.79	0.73	0.76
Weighted Avg	0.75	0.74	0.74
Samples Avg	0.71	0.74	0.70

Table 7: Per-class metrics for EfficientNetV2-S with 4, 5, 6, 7 blocks unfrozen (100 epochs, threshold tuning).

The model achieved excellent results on both common genres and rarer ones, showing its capacity to adapt both general and specialized visual features. The results are actually significantly better than anything obtained before in this project. The model is fast, its efficient, and its quite a bit more accurate than any of the previous ones. Our B0 model was really good, but even comparing this to that is a big jump. Some of these actually almost double in terms of the F1 like Biography, Mystery, War. The high performing ones like Animation is near perfect at 0.95. The rare genres or low performing ones like War, Sport, Mystery which were all around 0.2-0.3 in the Resnet50 model are all approaching 0.7-0.8 levels. These results here are of course threshold tuned as well, pre tuning, its around a couple percentage points less, but similar numbers as expected. We see both the recalls and precisions increased by a large amount, which means the F1 scores also did. It is seen that only 3 genres are inside the 0.6-0.69 region, the others are all above 0.7 for the F1 scores which is quite impressive considering that this is a multi-label classification problem with 19 genres.

Looking at the averaged results, it is seen that Micro F1 is 0.74 which is great, this shows that

the imbalance is not a big problem in terms of the results, similarly Macro F1 0.76 shows that it is good for both common and rare genres. In general these averaged results just show that the results are balanced and not skewed too much. Genres like Animation is totally still on a tier of its own, but that's expected since the Animation posters will almost definitely be different than any other genre, its unique so the model recognizes it well. Some rare genres surprisingly performed well like War, Sport, etc. This is something that was desired initially, this might have been showing the success of posweights and threshold tuning improvements we thought of at the start. Some of these turned out to have mediocre results, but I think that is due to the nature of their posters, opposite to the uniqueness of Animation posters. For example Thriller seems to be the lowest here, but that might be because its usually not a distinctive genre, and actually overlapping a lot with Action/Drama/Crime. Similar comments could be made about any of the other genres, like Adventure also one of the lower ones, overlaps a lot with Fantasy, Action, etc. But that was not the intention of the project. There are some visuals and plots that can be shown for this model as well such as this:

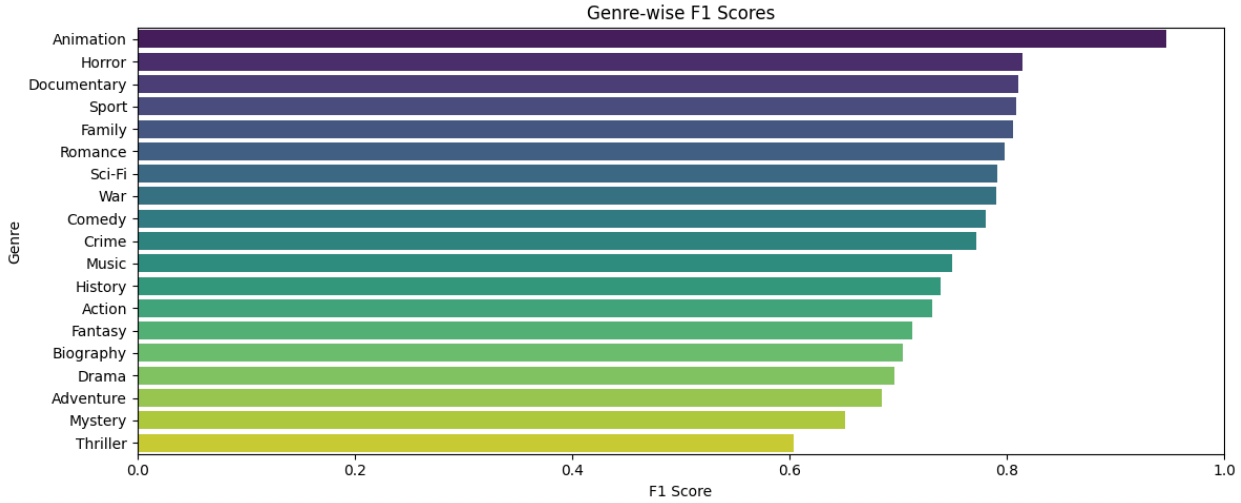

Figure 8: F1 Score visualization per genres for V2-S Model

This is a good way of visualizing the F1 scores of the genres, obviously the animation genre seems to be quite a bit ahead. But the good thing to see for this is how much better all the other genres look in comparison to the Resnet50 scores we had previously. There is actually not much else to discuss here that wasn't already discussed for the previous parts, just the fact that this model is significantly better than the other models. The Co-occurence map and other plots can be seen in the notebook but those are mostly redundant data. The data, weights for the simulation are also saved in the drive file, ready to use upon loading them and fact checking, the results do not change, they give these results, except the Efficient-B0 results overrode. Another interesting image to see is the prediction results of each of these for the model, whether it guessed 1 or 0 and what the truth was:

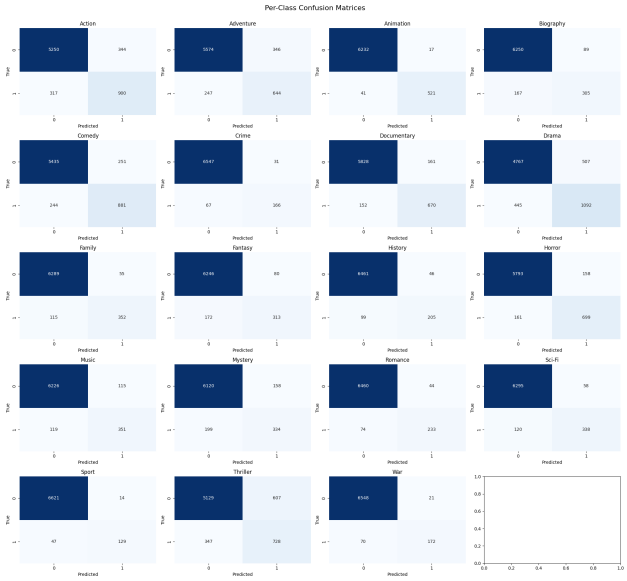

Figure 9: Per class Confusion Matrices for V2-S

This shows what the model did and its actually seen how it did a lot better than the Resnet50 model. Another one shows the Recall and Precision Map:

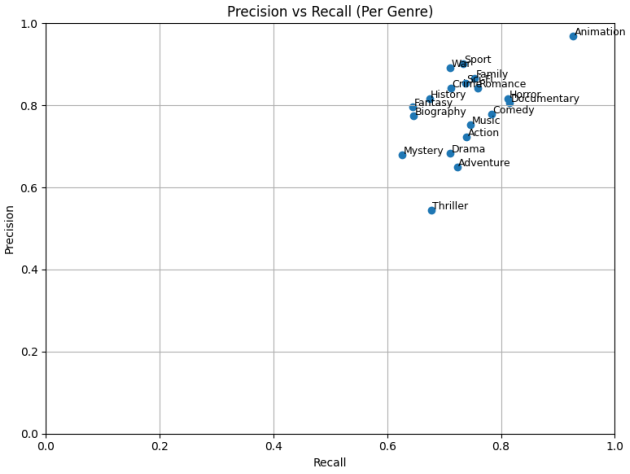

Figure 10: Recall and Precision Map of the Genres

It is easier to see what performed well and what didn't like this, top right is what we are looking for and bottom left is what we are avoiding, it seems that for this model they all ended up close to that top right area, with Animation leading the way at almost near perfect top right corner, and Thriller close to middle right. In general, this has been a great model that is definitely a huge success.

5.4 VGG16 Results - Historical Comparison

Initially, it was desired to test out VGG-16, but it is actually quite old as a model and almost certainly guaranteed to be not great. This model was created way before the other models used, it

is expected that the model would not only get worse results but also take a significant amount of computation time due to its larger nature.

Table 8: Classification Report Comparison for VGG16: Frozen vs Fine-Tuned

Genre	**VGG16 (No Tune)**			**VGG16 (Threshold-Tuned)**		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Action	0.59	0.23	0.33	0.42	0.53	0.47
Adventure	0.60	0.05	0.10	0.30	0.55	0.38
Animation	0.88	0.64	0.74	0.79	0.71	0.74
Biography	0.00	0.00	0.00	0.13	0.53	0.21
Comedy	0.70	0.25	0.37	0.50	0.56	0.53
Crime	0.36	0.02	0.03	0.34	0.25	0.29
Documentary	0.47	0.12	0.19	0.33	0.49	0.40
Drama	0.67	0.21	0.32	0.47	0.53	0.50
Family	0.67	0.03	0.05	0.29	0.48	0.36
Fantasy	0.00	0.00	0.00	0.15	0.46	0.22
History	0.00	0.00	0.00	0.10	0.18	0.13
Horror	0.54	0.29	0.38	0.38	0.62	0.47
Music	0.00	0.00	0.00	0.18	0.29	0.22
Mystery	0.00	0.00	0.00	0.16	0.41	0.23
Romance	0.64	0.02	0.04	0.38	0.42	0.40
Sci-Fi	1.00	0.00	0.01	0.21	0.42	0.28
Sport	0.00	0.00	0.00	0.11	0.02	0.04
Thriller	0.45	0.01	0.02	0.28	0.64	0.39
War	0.00	0.00	0.00	0.10	0.27	0.15
Micro avg	0.65	0.14	0.23	0.30	0.50	0.37
Macro avg	0.40	0.10	0.14	0.30	0.44	0.34
Weighted avg	0.49	0.14	0.19	0.34	0.50	0.40
Samples avg	0.22	0.15	0.17	0.32	0.50	0.36

It seems like the results of the VGG16 is really rough, this shows the problems with the model. It's one of the older models, around 11 years ago it was created, it is not efficient, it is too big and certainly not great when it comes to these problems. The no-tuned version seems to completely miss some rare genres, the averages are actually balanced by good genres like Animation, and since its imbalanced, the average F1 values look better than what they are. The rare genres are completely wrong, the model just missed them entirely. The tuned version definitely sees the biggest jump for this model, but even the best of this model seems to be struggling between 0.3-0.4 on average, and around 0.1-0.2 for the rare genres. In fact genres like Sport gave us 0.04, which is actually lower than a guess for 19 genres which is 1/19 which equals to more than 0.05. Either way, this is a good historical comparison, and shows just how far the current models have come. The model has a lot of parameters, its deep, but as it is known, that is not the criteria for success for all cases. Although of course, either way, the model did give some decent scores for especially the common genres, there are quite a bit of genres with 0.4-0.5 or more F1 scores. With more tuning, and epochs, it is possible that more scores could be squeezed out.

5.5 Model Comparison and Summary

To compare all models evaluated in this study, we summarize their peak validation metrics in Table 9. The F1 scores were compared previously but this is a more clear comparison

Table 9: Validation Results for All Models (Best Epoch with Threshold Tuning)

Model	Macro F1	Micro F1	Samples F1
VGG-16 (Historical Comparison)	0.34	0.37	0.36
ResNet50 (Baseline, 12 Epochs)	0.44	0.48	0.45
ResNet50 (Balanced, 50 Epochs)	0.43	0.46	0.43
EfficientNet-B0 (30 epochs)	0.53	0.55	0.52
EfficientNetV2-S (Blocks 6–7, Lower Epoch)	0.43	0.46	0.43
EfficientNetV2-S (Blocks 4–7, 100 Epoch)	**0.76**	**0.74**	**0.70**

Observations. The VGG-16 model, as expected, didn't give a great performance. It's an older model, its a large, inefficient model with known issues, but even with all that it had some decent results to show for historical comparison. The baseline ResNet50 model performed reasonably well, but adding `pos_weight` improved its ability to handle class imbalance, the rarer genres did better, but that did in turn give worse results for the common ones, so it was a matter of preference for choosing between the 2, especially considering the increased computational cost of the Balanced Model, but these considerations didn't matter at the end since they were both outperformed by better models later. Transitioning to EfficientNet architectures led to noticeable gains in all fronts really. The B0 model that was smaller and older did give good results considering everything, certainly a good improvement over the Resnet50 models. But the the V2-S model which is newer, faster and bigger, gave much better results, and a significant jump over the models. Of course, during all of this tuning like threshold tuning as it was explained carried an important role of getting extra performance. The averaged results in the table don't tell the whole story, as it was explained in detail previously, each model showed their scores for genres before. Extra plots, outputs, and logs are all available in the notebook, the drive files can be accessed to load the weights and simply see the reports without re-running them, especially for the best one, the V2-s model.

6 Conclusion, Failures, Future Work

In this project, we explored several deep learning architectures for multi-label movie genre classification using poster images. Starting from VGG-16 we saw what an older model would produce as results for this difficult multi-label problem, it gave the results as expected. Later on, ResNet50 was used as a baseline, we progressively improved performance by addressing class imbalance with weighted loss functions and by employing more powerful architectures such as EfficientNet-B0 and EfficientNetV2-S.

The most effective setup was found to be EfficientNetV2-S with blocks 4 through 7 unfrozen and per-class threshold tuning, achieving a macro F1 score of 0.76, micro F1 of 0.74, and samples F1 of 0.70. These results demonstrate the value of fine-tuning deeper layers and tuning thresholds in imbalanced multi-label settings. The results are satisfactory results considering the multi-label aspect of it and how it includes 19 genres with imbalanced data.

There might be certain aspects of this that can be improved. One thing is the available software/hardware. Due to the limitations of it, the simulations ran into some issues. Google Collab

was used for this project due to lack of hardware and the subscription model for it was obtained, but even with the subscription model of the tool, the GPU ran into some limitations and stopped working. This caused the amount of runs that could be made. The simulation part was already detailed and took a lot of time, oftentimes running dozens of these for days cumulatively, so at the end considering all of these, as it is always the case, more simulations could have been done. One issue was saving the data inside Collab, during the simulations, problem with it occurred, first it was realized that the data reset, then even saving it directly didn't work, and also during some runs the code for saving it simply didn't work resulting in a waste of time, later a right way of saving it using drive was utilized.

The models used for this were picked well, but again, as it is always the case, better models could have been found, maybe newer generation but less commonly used models could have been used, or transformers could have been utilized but the proposal didn't consider all of that to begin with. These are mostly the things that can be considered in the future, other models can be utilized basically. And also considering that, creating a new model from scratch is of course always an option, after understanding how these models work, it is certainly possible to make one but also unlikely to create a better model considering their complexities.

In general, the results can naturally be better than what was found here, but we think the obtained result of the V2-S is a big success considering the problem. weighted loss solutions and threshold tuning were vital parts of improving the results.

Throughout the experiment, what felt like a complex project of handling a multi-label classification problem for predicting movie genres using posters was made. Of course for this, handling the data correctly was a tough part to handle, many aspects of it has to be considered, like the data being processed properly, corresponding to the right things, not having duplicates or wrong classifiers, transforming it in a way that can be used for this multi-label issue, etc. The entire process of that was learned. Then using transfer learning was learned in detail, this process includes picking the right models, writing all the necessary codes, handling the right metrics to analyze, correctly understanding what data to look at and what exactly do they tell us, processing these results, understand what needs to be done further on, how to tune these complex models for these complex problems and many more. These are of course, surface level explanations for the Conclusion part, but what what was learned throughout the project, can be seen from the work and the report in much more detail. The learnings of this project will be utilized later in our respective personal projects, at least in the form of using transfer learning and Neural Networks paired up with the desired datasets such as using Fluid Mechanics direct numerical simulation data to come up with turbulent flow modelling and more.

6.1 Appendix: Thresholds Used

The following thresholds were used for binarizing genre probabilities in the final EfficientNetV2-S (blocks 4–7) model(more plots and analysis and logs can be seen in the notebook):

Genre	Threshold
Action	0.30
Adventure	0.45
Animation	0.65
Biography	0.50
Comedy	0.30
Crime	0.20
Documentary	0.35
Drama	0.30
Family	0.45
Fantasy	0.15
History	0.10
Horror	0.10
Music	0.40
Mystery	0.25
Romance	0.15
Sci-Fi	0.25
Sport	0.75
Thriller	0.20
War	0.80

Table 10: Per-class optimal thresholds derived from validation data.

Supplementary Material

Code repository and training logs are available at: `https://github.com/yourusername/movie-genre-classific`

References

[1] Gabriel Barney and Kris Kaya. Predicting genre from movie posters. CS229 Course Project, Stanford University, 2022. `https://github.com/barneyga/229_pythonfiles`.

[2] Wei-Ta Chu and Hung-Jui Guo. Movie genre classification based on poster images with deep neural networks. In *Proceedings of the Workshop on Multimodal Understanding of Social, Affective and Subjective Attributes (MUSA2)*, pages 39–45, 2017.

[3] Mostafa Dewidar. Inferring movie genres from their poster. CS231n Course Project, Stanford University, 2022. `https://cs231n.stanford.edu/2022/reports.html`.

[4] Marina Ivašić-Kos, Miran Pobar, and Luka Mikec. Movie posters classification into genres based on low-level features. *MIPRO, IEEE Conference Proceedings*, pages 1198–1203, 2014.

[5] Kaushil Kundalia, Yash Patel, and Manan Shah. Multi-label movie genre detection from a movie poster using knowledge transfer learning. *Augmented Human Research*, 5(11), 2020.

[6] Sungkyu Loves. Movie posters by genre. `https://www.kaggle.com/datasets/lovesunkyu/movie-posters-by-genre`, 2019. Accessed: 2025-05-20.

[7] Zeeshan Rasheed and Mubarak Shah. Movie genre classification by exploiting audio-visual features of previews. In *IEEE International Conference on Pattern Recognition*, pages 1086–1089, 2002.