# Predict genre of the book by its cover


# By Sayali Walke


Code- https://github.com/sayaliwalke30/JudgeBookByCover

### Problem Statement:

Book covers communicate information to potential readers, but can that same information be learned by computers? The goal is to determine if genre information can be learned based on the visual aspects of a cover created by the designer.

Books come with a wide variety of book covers and styles, including nondescript and misleading covers. Books come with a wide variety of book covers and styles, including nondescript and misleading covers. Books come with a wide variety of book covers and styles, including nondescript and misleading covers.

### Approach:

To tackle this problem I used the concept of transfer learning and developed a Convolutional Neural Networks (CNN) based system for book cover genre classification.

I also developed simple CNN model but the results were very unsatisfactory. So, I used a pre-trained network on ImageNet. By pre-training VGG16 and ResNet50 models on a very large dataset such as ImageNet, it's possible to take advantage of the learned features and transfer it to other applications.

## Steps Followed:

### 1] Web Scraping

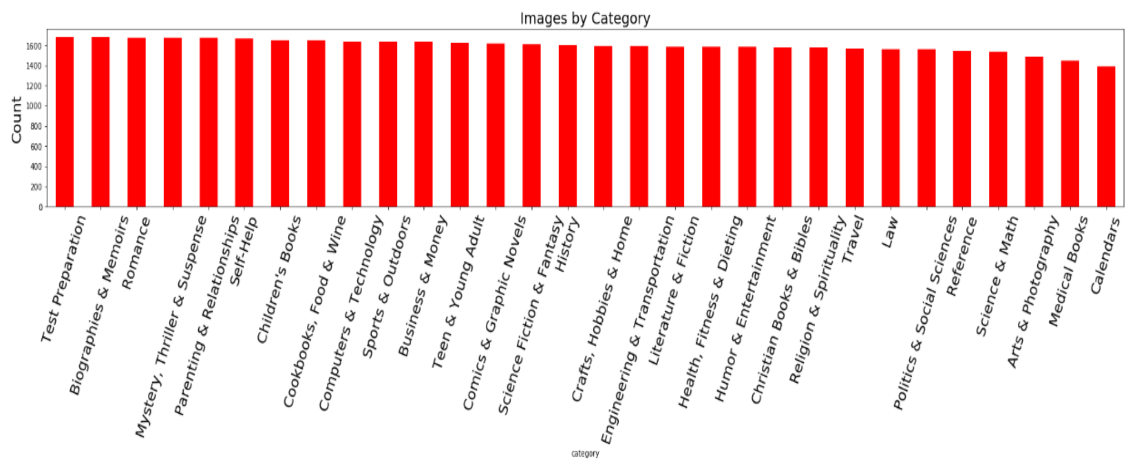
Script to download data-
https://github.com/sayaliwalke30/JudgeBookByCover/blob/master/WebScrappingBookData.ipynb

The dataset for this task is downloaded from Amazon.com

The dataset contains 57,000 book cover images divided into 30 classes.

**The data distribution looks like ->**

Out[90]: Text(0.5, 1.0, 'Images by Category')




In [12]:
```
df.groupby(['Category']).count()
```

Out[12]:

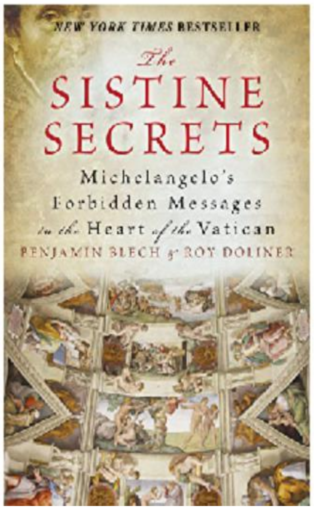
| | Amazon ID (ASIN) | Filename | Image URL | Title | Author | Category ID |
|---|---|---|---|---|---|---|
| **Category** | | | | | | |
| Arts & Photography | 1487 | 1487 | 1487 | 1487 | 1487 | 1487 |
| Biographies & Memoirs | 1679 | 1679 | 1679 | 1679 | 1679 | 1679 |
| Business & Money | 1629 | 1629 | 1629 | 1629 | 1629 | 1629 |
| Calendars | 1388 | 1388 | 1388 | 1388 | 1388 | 1388 |
| Children's Books | 1653 | 1653 | 1653 | 1653 | 1653 | 1653 |
| Christian Books & Bibles | 1579 | 1579 | 1579 | 1579 | 1579 | 1579 |
| Comics & Graphic Novels | 1617 | 1617 | 1617 | 1617 | 1617 | 1617 |
| Computers & Technology | 1633 | 1633 | 1633 | 1633 | 1633 | 1633 |
| Cookbooks, Food & Wine | 1651 | 1651 | 1651 | 1651 | 1651 | 1651 |
| Crafts, Hobbies & Home | 1594 | 1594 | 1594 | 1594 | 1594 | 1594 |
| Engineering & Transportation | 1593 | 1593 | 1593 | 1593 | 1593 | 1593 |
| Health, Fitness & Dieting | 1586 | 1586 | 1586 | 1586 | 1586 | 1586 |
| History | 1598 | 1598 | 1598 | 1598 | 1598 | 1598 |
| Humor & Entertainment | 1584 | 1584 | 1584 | 1584 | 1584 | 1584 |
| Law | 1562 | 1562 | 1562 | 1562 | 1562 | 1562 |
| Literature & Fiction | 1588 | 1588 | 1588 | 1588 | 1588 | 1588 |
| Medical Books | 1442 | 1442 | 1442 | 1442 | 1442 | 1442 |
| Mystery, Thriller & Suspense | 1670 | 1670 | 1670 | 1670 | 1670 | 1670 |

## 2] Data Exploration

Code- https://github.com/sayaliwalke30/JudgeBookByCover/blob/master/BookDataExplore.ipynb

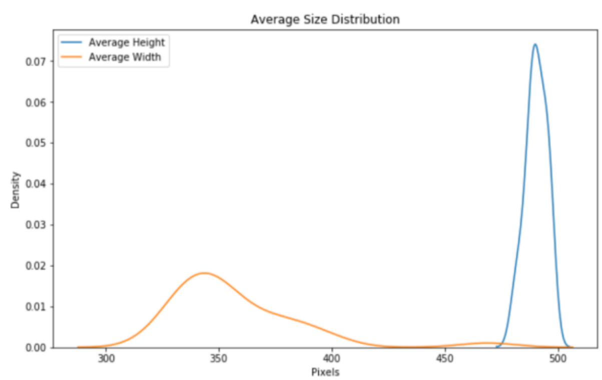
**Cover image looks like ->**

```
In [40]: # Example image
         x = Image.open(data_dir + 'Arts & Photography/006146905X.jpg')
         np.array(x).shape
         imshow(x)
```




**Height and width of the images looks like this ->**

```
Out[84]: Text(0.5, 1.0, 'Average Size Distribution')
```




**I have also calculated Mean and Standard deviations of images dataset for RGB channel to normalize the image**

```
In [156]: # Since the
          bookdata_stats = [[redMean/255, greenMean/255, blueMean/255], [redStd/255, blueStd/255, greenStd/255]]
          print(bookdata_stats)

          [[0.4674833757541233, 0.5044437399038161, 0.5275549351844859], [1.3134155500344147, 1.3902920492485198, 1.
          289612176433768]]
```

However, for transfer learning I used image net standards because I got better results using standard values. Also, I read few articles and they suggested imagenet_stats, not the dataset stats, should be applied whenever using a model that was pretrained with imagenet.

## 3] Image Augmentation

To build a powerful image classifier using very little training data, image augmentation is usually required to boost the performance of deep networks

**Image augmentation** artificially creates training images through different ways of processing or combination of multiple processing, such as random rotation, shifts, shear and flips, etc.

For increasing training and validation dataset images I used random rotation, random horizontal flips.

## 4] Image Transformation

The dataset that I downloaded from Amazon had different size of images so, I performed center cropping on dataset and reshape them to 224 x 224. This is the size of Image net images and is therefore what the model expects. The images that are larger than this will be truncated while the smaller images will be interpolated. Also I normalized the RGB channel values of images by using imagenet_stats.

## 5] CNN from scratch

Code- -
https://github.com/sayaliwalke30/JudgeBookByCover/blob/master/CNNBookByCover%20(1).ipynb

- First, I started with image classification using a simple CNN
- CNN stands for Convolutional Neural Network, where each image goes through a series of convolution and max pooling for features extraction.
- With so many images, it took almost 8 hours to train the model, and achieved an accuracy of 23% on test data

## 6] Transfer Learning

Code - https://github.com/sayaliwalke30/JudgeBookByCover/blob/master/judgeBookByCover.ipynb

Most categories only have 1500 images which typically isn't enough for a neural network to learn to high accuracy. Therefore, after training a CNN from scratch, I used a pre-built and pre-trained model applying transfer learning.

The basic premise of transfer learning is simple: take a model trained on a large dataset and *transfer* its knowledge to a smaller dataset. For object recognition with a CNN, we freeze the early convolutional layers of the network and only train the last few layers which make a prediction.

**I followed following steps in transfer learning:**

- Load in a pre-trained CNN model trained on a large dataset
- Freeze parameters (weights) in model's lower convolutional layers
- Add custom classifier with several layers of trainable parameters to model
- Train classifier layers on training data available for task
- Fine-tune hyperparameters and unfreeze more layers as needed

For this problem I used **VGG16** and **ResNet50** pre-trained models. Pretrained models like VGG-16 and ResNet50 are already trained on ImageNet which comprises of disparate categories of images. As the model is trained on huge dataset, it has learned a good representation of low level features like spatial, edges, rotation, lighting, shapes and these features can be shared across to enable the knowledge transfer and act as a feature extractor for new images in different computer vision problems. These new images might be of completely different categories from the source dataset, but the pretrained model should still be able to extract relevant features from these images based on the principles of transfer learning

## 7] Results

Results

**The table shows the accuracy in percentage on the test data for each model**

| CNN | 23 % |
|---|---|
| VGG16 | 59.375 % |
| ResNet50 | 62.5 % |

## Results

Accuracy for a prediction and a target in terms of topk for each category:

**ResNet50**

| | class | top1 | top5 | loss |
|---|---|---|---|---|
| 0 | Biographies & Memoirs | 69.062500 | 100.0 | 0.916479 |
| 1 | Mystery, Thriller & Suspense | 60.322581 | 100.0 | 1.104819 |
| 2 | Parenting & Relationships | 57.200000 | 100.0 | 1.142326 |
| 3 | Romance | 53.666667 | 100.0 | 1.167043 |
| 4 | Test Preparation | 72.666667 | 100.0 | 0.753208 |

So we can see that the category Romance has the least accuracy

**VGG16**

| | class | top1 | top5 | loss |
|---|---|---|---|---|
| 0 | Biographies & Memoirs | 50.937500 | 100.0 | 1.199545 |
| 1 | Mystery, Thriller & Suspense | 66.451613 | 100.0 | 1.045316 |
| 2 | Parenting & Relationships | 54.400000 | 100.0 | 1.129545 |
| 3 | Romance | 57.666667 | 100.0 | 1.151498 |
| 4 | Test Preparation | 70.666667 | 100.0 | 0.776581 |

So we can see that the category Biographies & Memoirs has the least accuracy

## 8] Conclusions

- Image classification can be done using neural network models. Identifying patterns and extracting features on images are what deep learning models can do, and they do it very well. So, I got best possible accuracy of 62% using Resnet50.
- I will explore different pre-trained problems in future like Alexnet.
- I will also experiment on tuning the model performance.