Spring 2020

Book Genre Classification By Its Cover Using A Multi-view Learning Approach

Chandra Shakhar Kundu
Western Kentucky University, chandra.kundu537@topper.wku.edu

Recommended Citation

Kundu, Chandra Shakhar, "Book Genre Classification By Its Cover Using A Multi-view Learning Approach" (2020). *Masters Theses & Specialist Projects.* Paper 3205.
https://digitalcommons.wku.edu/theses/3205

BOOK GENRE CLASSIFICATION BY ITS COVER USING A MULTI-VIEW LEARNING APPROACH

A Thesis
Presented to
The Faculty of the Department of Mathematics
Western Kentucky University
Bowling Green, Kentucky

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science

By
Chandra Shakhar Kundu

May 2020

BOOK GENRE CLASSIFICATION BY ITS COVER USING A MULTI-VIEW LEARNING APPROACH

Date Recommended 06/26/2020

Lukun Zheng Digitally signed by Lukun Zheng
Date: 2020.06.28 17:47:54 -05'00'

Dr. Lukun Zheng, Director of Thesis

Mark Robinson Digitally signed by Mark Robinson
Date: 2020.06.28 15:08:37 -05'00'

Dr. Mark Robinson

Ngoc Nguyen Digitally signed by Ngoc Nguyen
Date: 2020.06.26 21:21:35 -05'00'

Dr. Ngoc Nguyen

[illegible]

Dr. Qi Li

Cheryl D Davis Digitally signed by Cheryl D Davis
Date: 2020.06.29 09:02:10 -05'00'

Dean, Graduate School Date

ACKNOWLEDGMENTS

I would like to thank my thesis advisor Dr. Lukun Zheng, for his patience, motivation, and enthusiasm. He guided me to deal with technical challenges and provided me useful hints which helped me improve the quality of this work.

Besides my advisor, my most profound gratitude goes to the rest of my thesis committee, Dr. Mark Robinson, Dr. Ngoc Nguyen, and Dr. Qi Li for their insightful comments, advice, and support.

I am grateful to my parents and family who have been encouraging me in countless ways. I am particularly thankful to my sweetheart Mitu, not only for her endless patience but also for her positive nature that increased my motivation in many circumstances.

Finally, my sincere thanks go to the faculty at WKU for investing your time in me and giving me an opportunity to gain teaching experience.

Contents

List of Figures

List of Tables

BOOK GENRE CLASSIFICATION BY ITS COVER USING A MULTI-VIEW LEARNING APPROACH

Chandra Kundu May 2020 43 Pages

Directed by: Dr. Lukun Zheng, Dr. Mark Robinson, Dr. Ngoc Nguyen & Dr. Qi Li

Department of Mathematics Western Kentucky University

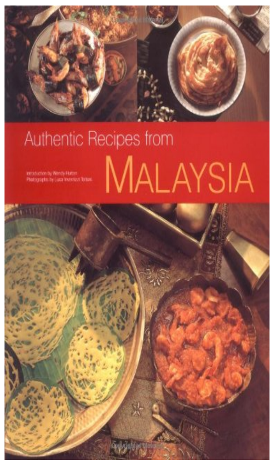
An interesting topic in the visual analysis is to determine the genre of a book by its cover. The book cover is the very first communication to the reader which shapes the reader's expectation about the type of the book. Each book cover is carefully designed by the cover designers and typographers to convey the visual representation of its content. In this study, we explore several different deep learning approaches for predicting the genre from the cover image alone, such as MobileNet V1, MobileNet V2, ResNet50, Inception V2. Moreover, we add an extra modality by extracting text from the cover image. We explore some text classification algorithms such as LSTM and Universal Sentence Encoder. Moreover, we focus on multi-modal fusion based on two best performing models on cover images and text. Finally, we propose the use of Deep Canonical Correlation Analysis (DCCA) to jointly learn the features from two modalities: image and text, then use a support vector machine classifier to predict the genre of the book. Overall, the concatenation of two models without DCCA yields the best result. However, our analysis revealed the weakness of these models for solving this task on our used dataset. Our results suggest that solving this task to a satisfactory level needs significant efforts and a much-more accurate dataset.

Chapter 1

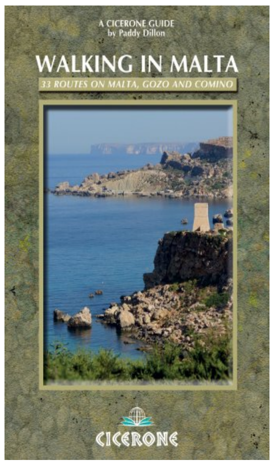
Introduction

Image classification is a process in computer vision to classify an image according to its visual content. While classifying an image is trivial for humans, robust image classification is still a challenge in computer vision applications. To classify an image, a necessary step is the extraction of the visual attributes.

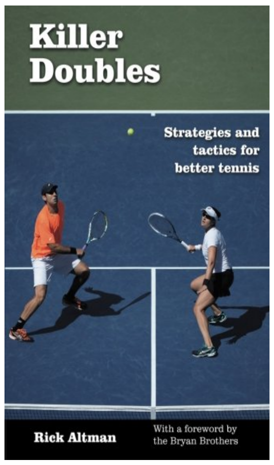
When studying visual attributes for different types of images, we found that book covers are very special types of visual representations that are designed to attract people to give time and money to read the corresponding book. To discourage being judgmental to someone by the appearance, we often say the idiom, don't judge a book by its cover. However, a reader often chooses a book by its cover. Information and objects on a book cover, therefore, should be attractive and relative to readers' tastes and choices.



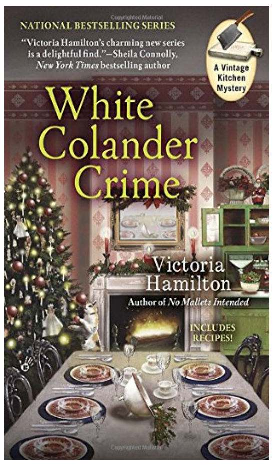
(a)



(b)


Killer Doubles
Strategies and tactics for better tennis
Rick Altman
With a foreword by the Bryan Brothers

(c)



(d)

Figure 1.0.1: Sample book covers. The genres are: (a) Cookbooks, Food & Wine, (b) Travel, (c) Sports & Outdoors and (d) Mystery, Thriller & Suspense

As we can see in the **Figure 1.0.1**, most of the covers present the most important imagery of the corresponding book. For example, **Figure 1.0.1(a)** shows different kinds of good looking food which is attractive to people who like books on the *Cookbooks, Food & Wine* genre. Besides, different books target different kinds of readers, and covers should concisely present genre information. For example, we can recognize that **Figure 1.0.1(a)** presents food elements, **Figure 1.0.1(b)** presents a beautiful scenery and **Figure 1.0.1(c)** presents some sports activity and **Figure 1.0.1(d)** seems mysterious. Moreover, book covers usually present important objects like the author's name, book titles, short details of the book, etc. Overall, the first aforementioned observation is highly related to image aesthetics or attractiveness estimation; the second observation is related to the genre; while the third observation is involved with object detection.

The main problem of book genre classification is that book genres are not precisely defined. Another problem, because a book can be part of multiple genres, it is troublesome, even for a human without reading the book, to categorize it into a single category. Besides, many book covers are not designed carefully, thus contains misleading cover information. Therefore, we think the book cover is a good research option with many technical challenges. To overcome these difficulties, we consider the text, extracting from cover images as another modality to perform this task. This study pays attention to state-of-art models for image classifications. Several contributions are made in this study, as follows:

- We explore MobileNet V1 [29], MobileNet V2 [30], ResNet50 [28], Inception V2 [24] to classify books based on cover image alone. Moreover, we employ LSTM [32] and Universal Sentence Encoder [33] to predict the book category based on text extracted from the cover image.

- We choose two best models from the aforementioned models and use those in a multi-modal fusion. In addition, we employ Deep Canonical Correlation

Analysis(DCCA) [38] for book genre classification intending to learn joint representations between book covers and cover text. To the best of our knowledge, this work is the first study where a DCCA architecture with two-branch neural networks for image and text is studied for book genre classification.

- We analyze our results and outline the challenges due to which even the state-of-the-art models fail to solve this problem

This study can aid the design process by providing the underlying information about cover images, and help in the promotion and automatic categorization of books. Automatic categorization of books based on covers without explicit human intervention could notably improve the performance of book retrieval systems. This study could also be extended to other domains, such as classification of movie genre, game genre, and music albums.

The rest of this paper is organized as follows. Chapter 2 provides a brief survey of the previous work on genre classification as well as book genre classification. Chapter 3 describes the preliminaries of deep learning. The proposed methodology to classify books based on the cover is given in Chapter 4. In chapter 5, we discuss evaluation, results, and analysis. Finally, we conclude the paper with the concluding remarks in chapter 6.

Chapter 2
Related Works

In recent years, there has been an increasing interest in automated genre classification based on images by leveraging the strength of the deep neural network. There are multiple attempts to classify movie genre based on its poster with deep neural network [6, 4]. Low-level features such as the dominant color of the whole poster, texture, or color histogram have been used to find the genre of movies [2, 3]. Convolutional neural networks have been used to categorize the genre of paintings and artworks [1, 7, 12]. There are similar works done in music album classification [8, 9].

In book genre categorization, the very first approach using deep neural networks was based on only its cover. The pretrained LeNet [19] and AlexNet [21] architectures have been used to find the genre of a book by its cover. A dataset consisting of 30 categories has also been published in that study [5]. Due to the complexity of the dataset, the study achieved a baseline accuracy of 24.5%. In [10], the authors used the same dataset but chose only 10 of the original genres. They applied VGG16 [22], SqueezeNet [25] and ResNet [28] and achieved higher accuracy than that of [5]. In [11], the authors developed a neural network consists of 3 convolutional layers, each followed directly by a 2x2 max-pooling layer with non-overlapping windows. They worked on a completely different unpublished dataset, crawled from Goodreads.com, comprising of 14 categories.

The aforementioned studies on book genre classification have focused on book cover images in a "single-view" setting. In our work, we first try to investigate deep learning approaches concatenating the cover image and textual information on the image. Then we focus on a multi-view deep learning approach called DCCA [38]. DCCA uses multiple stacked neural network layers of nonlinear transformations to simultaneously learn the representations of two views of data that are maximally correlated to transform. DCCA has been used to classify Greek folk music using lyrics and audio [39]. Some studies have found that DCCA is highly efficient in recognizing emotion from multi-modal signals [41, 40].

In our case, we use image and text features as the two modalities into hyperspace by using specified canonical correlation analysis constraints to simultaneously learn the representations of two modalities that are maximally correlated. We compare it with [5] and show the performance of different variants of the proposed framework.

Chapter 3
Preliminaries

We begin this chapter by briefly discussing the theoretical background relevant to Deep Learning, which is the primary technique used in the study.

3.1 Deep Neural Networks

A neural network consists of single neurons that are connected with each other. If a single neuron takes multiple inputs x_i, it computes the output as follows:

$$output = f(\sum_{i=1}^{n} w_i x_i + b) \tag{3.1.1}$$

where n is the number of inputs, the w_i are defined as weights, b as bias and $f(\cdot)$ is a nonlinear activation function. Often, we call it is a single-layer perceptron. The output of one neuron can be the input of another neuron. In a multi-layer perceptron (Figure 3.1.1), the neurons are grouped into layers, each layer is fully connected to the next one (each neuron of a layer is connected with each neuron of the subsequent layer) and the connections do not form any closed directed cycles [13].

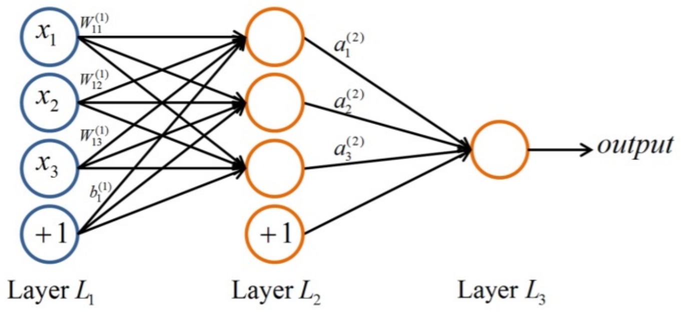


Figure 3.1.1: Example for a feed-forward neural network. [14]

In Figure 3.1.1, the leftmost layer is called input layer, the rightmost layer is called output layer. The layer in the middle is denoted as hidden layer, because its values are not observed in the training set. The $i-th$ layer is labeled as L_i. Thus, the input layer is defined as L_1, the second (hidden) layer as L_2 and the third (output) layer as L_3. The parameters of the model are $(W,b) = (W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)})$, where $b_i^{(l)}$ is the bias corresponding to neuron i in layer $l+1$ and $W_{ij}^{(l)}$ denotes the weight parameter corresponding to the connection between the unit j in layer l and the neuron i in layer $l+1$. The activations of the hidden units are computed as follows:

$$a_1^{(2)} = f(W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + b_1^{(1)})$$

$$a_2^{(2)} = f(W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + b_2^{(1)})$$

$$a_3^{(2)} = f(W_{31}^{(1)}x_1 + W_{32}^{(1)}x_2 + W_{33}^{(1)}x_3 + b_3^{(1)})$$

where $a_i^{(l)}$ denotes the activation of unit i in layer l. The activations of the hidden units are used as the inputs of the next layer. The output is computed as follows:

$$output = a_1^{(3)} = \sigma(W_{11}^{(2)}a_1^{(2)} + W_{12}^{(2)}a_2^{(2)} + W_{13}^{(2)}a_3^{(2)} + b_1^{(2)})$$

where σ is the activation function for the output layer. This activation function can be different from that of the hidden units. This process is known as forward propagation[14]. A deep neural network can have multiple hidden layers and different numbers of units in each layer. The structures of connectivity between the neurons can be different and there could be more than one output in the case of multi-label classification.

3.2 Activation Function

The activation function can be both linear and non-linear. In this section, we will discuss different activation functions. It is important to note that the non-linear activation function is necessary for a deep network because the composition of consecutive linear transformations is itself only a linear transformation [13].

Linear Activation: This is the simplest form of activation function which is equivalent to having no activation at all.

$$\sigma(x) = x$$

Sigmoid Function: This is also know as logistic activation function. This function squashes the input to values between 0 and 1. This property is useful when the desired output is probabilistic value.

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Hyperbolic Tangent: The *tanh* non-linearity is similar to Sigmoid because it squashes the input between -1 to 1.

$$\sigma(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

ReLU: Rectified Linear Units(ReLU) is half-rectified function. The input of real numbers less than 0 is output zero and positive input remain as it is.

$$\sigma(x) = max(0, x)$$

Softmax: Softmax turns arbitrary real values into probabilities in the range [0, 1] and add up to 1.

$$\sigma(x) = \frac{e^{x_j}}{\sum_{j=1}^{n} e^{x_j}}$$

3.3 Learning in Neural Networks

From the universal approximation theorem, we know that a feed-forward artificial neural network with a single hidden layer containing a finite number of neurons can approximate continuous functions on compact subsets of $\mathbb{R}^n$, if the activation function is continuous, non-constant, bounded, and monotonically-increasing [15]. Generally, a neural network is capable of approximating large class functions by adjusting the parameters and the activation functions. The process of learning these parameters when we know the output values for a set of inputs values is called supervised learning. After learning these parameters, a neural network is able to predict the output for unlabeled inputs.

For the process of learning, we first choose a suitable activation function and then define a loss function or error function, whose value is optimized (maximized or minimized) by adjusting the neural network parameters. We want to have a loss function that is a convex with a continuous first derivative. This simplifies the optimization, which can be performed using gradient descent or related algorithms. We use $\theta = [W, b]$ to denote the parameters of the network, so we denote the loss function as $E(W, b) = E(\theta)$

For standard regression problems, the output-layer activation function is the linear activation function. Assuming that the training data set is given as $x_1, x_2, ..., x_n$, with the corresponding output vectors as $y_1, y_2, ..., y_n$ and the output of the network is defined as $f_{W,b}(x)$, the loss function can be defined as:

$$E(\theta) = \frac{1}{2}\sum_{i=1}^{n} ||f_{W,b}(x_i) - y_i||^2$$

which is known as the sum-of-squares error function.

For binary classification problems, the two classes are denoted as $0, 1$ and the activation function of the output unit is chosen to be the logistic sigmoid function. If $y = 1$ is defined as label for the class C1 and $y = 0$ denotes class C2, then the output $f_{W,b}(x)$ can be interpreted as the class membership probability for C1. The class membership probability for C2 is then given as $1 - f_{W,b}(x)$. In this case, the error function can be chosen as a cross-entropy error function:

$$E(\theta) = -\sum_{i=1}^{n}\{y_i \ln f(x_i) + (1 - y_i)\ln(1 - f(x_i))\}$$

In the case of multi-class classification problems, each input example is assigned to exactly one of k classes. Since every output unit is associated with one class, in a k-class classification problem, the network has k output units. Because each output unit is assigned to a specific class and the network outputs are interpreted as the class membership probabilities $(P(y = j|x)$, where $j = 1, 2, 3, ..., k$. The final prediction can be obtained by selecting the class with the highest probability allocated. For the multi-class classification problem, a reasonable error function is the multi-class cross-entropy error function:

$$E(\theta) = \sum_{i=1}^{n}\sum_{j=1}^{k} \delta_{y_{ij}} \ln \sigma_j$$

where σ_j is the output from the softmax activation function and $\delta_{y_{ij}}$ is the Kronecker delta [13].

3.4 Backpropagation

So far, we described the architecture of neural networks and the process to compute the output and loss. However, the key problem in the neural network is how to find suitable parameters (W, b) so that loss is optimized. We call this process as training. Though there are many ways to train a neural network, we focus on training a network using error backpropagation and gradient descent.

The target of training is to find the parameters which optimize the loss function. This cannot be obtained by finding an analytical solution for the equation $\nabla E(\theta) = 0$.[1] This is because the error function is a non-convex, which has many local minima beside the global minimum. Most techniques involve choosing an initial value for the parameter-vector $\theta^{(0)}$ and perform iterative steps to find a solution of the form:

$$\theta^{(k+1)} = \theta^{(k)} + \Delta^{(k+1)}$$

where k denotes the iteration step [13]. A single iteration step involves a small step in the direction of the negative gradient of the form:

$$\theta^{(k+1)} = \theta^{(k)} - \alpha \nabla E(\theta^{(k)})$$

where $\nabla E(\theta)$ points in the direction of the greatest rate of increase of the error function and $\alpha > 0$ is called learning rate. These steps of changing the parameters of the neural network are performed until a minimum is reached (preferably the global minimum). Too small a choice of α can lead to slow convergence of the model. Contrary, if the learning rate is too large, the gradient descent algorithm can diverge[16].

After each update step, it is necessary to compute $E(\theta)$ for the new parameter

[1]The nabla operator ∇ denotes the vector of the partial derivative operators

vector $\theta^{(k+1)}$. In the simple batch gradient descent method, every parameter update involves the whole training set being processed in order to evaluate $E(\theta)$. In contrast, the stochastic gradient descent method uses only one training example at a time to evaluate $E(\theta)$, where the data example is either chosen by cycling through the training set in sequence or selecting at random. Another approach is mini-batch gradient descent, where $E(\theta)$ is evaluated by using B data examples.There are different kinds of gradient descent optimization algorithms used in popular deep learning frameworks that have been described in [17].

3.5 Overfitting and Walk Around

If the network is trained for a long time, it may perform extremely well with the training data. However, doing so could increase the error for the test data. This is known as the problem of overfitting. One way to combat the problem of overfitting is to choose the number of hidden units just large enough and add a regularization term to the error function [16]. This added term to the acts as a mechanism for weight decay.

$$\widetilde{E}(\theta) = E(\theta) + \lambda \sum_i (\theta_i)^2$$

where λ is called regularization parameter and it reduces the variance while the bias is not overly increased. Adding a regularization term of the form $\lambda \sum_i (\theta_i)^2$ is called L2-regularization, whereas in L1-regularization the term $\lambda \sum_i |\theta_i|$ would be added to the error function.

Early stopping is another way to control the complexity of the model and to avoid overfitting [16]. The idea behind early stopping is to use a validation set to measure the performance of the model during the training process. In training, the value of the loss function is decreased in each step. The value of the loss function is called a validation error. It will keep reducing, however, after certain steps, this error will start increasing due to the overfitting on the training data. The idea here is to stop the training prior to convergence best on the lowest validation error.

Dropout is another technique that can be used to avoid overfitting. The deeper a network is, the higher the number of the parameters gets. Due to the fact of having a large number of parameters, the model tries to remember the train data rather than learning it. This also causes an overfitted model. Dropout means dropping a certain number of connections in the network during the training. This prevents the network from favoring certain connections all the time [18].

3.6 Convolutional Neural Network

In this section, we will discuss the basics of the Convolutional Neural Network (CNN) which is the next stage of the evolution of the neural network. CNNs were introduced as a variant of vanilla neural networks modeled after the visual cortex of animals [19]. The general motivation was that the visual cortex in mammals consists of layers of simple cells and complex cells, and as an image is being processed through the cortex, progressively richer features of the image are detected. Similarly, CNNs consist of stacks of convolutional layers just like the simple cell and pooling layers which can be thought of as the complex cells. These networks have been shown to learn high-level features in the form of filters in convolutional layers. Each layer of a convolution neural network consists of a set of learnable filters, a non-linear activation, and possibly a pooling operation. Here, convolution is the key operation. Suppose, $F = (v_{ij}) \in \mathbb{R}^{n \times n}$ is a filter and $I = (i_{ij}) \in \mathbb{R}^{n \times n}$ is a patch of an image, the convolution of the filter and the image patch is defined as:

$$F \cdot I = \sum_{j=1}^{n} \sum_{k=1}^{n} v_{jk} i_{jk}$$

The main concept of CNN is to slide the filter over the image, convolving the filter with each patch of the image as it slides. The number of pixels by which the filter moves each time its slides can be changed by specifying a stride. Each convolution provides an output and the resulting outputs of all of the convolutions of a filter with a whole image produce a feature map. A non-linear activation is then applied element-wise to the resulting feature map. Finally, the feature map can be passed through a pooling operation. The pooling layer is used to down-sample the feature map in height and width. The most common pooling method is known as max-pooling.

3.7 Convolutional Architectures

In this section, we briefly discuss a few CNN Architectures that comprise the current state-of-art.

3.7.1 LeNet

The very first kind of successful CNN application is LeNet (Figure: 3.7.1). It was developed to read zip codes and digits. The most known LeNet-5 consists of a sequence of 3 layers: convolution, pooling, non-linearity [19]. Inputs are normalized using mean and standard deviation to accelerate training. Sparse connection matrix between layers to avoid the large computational cost.

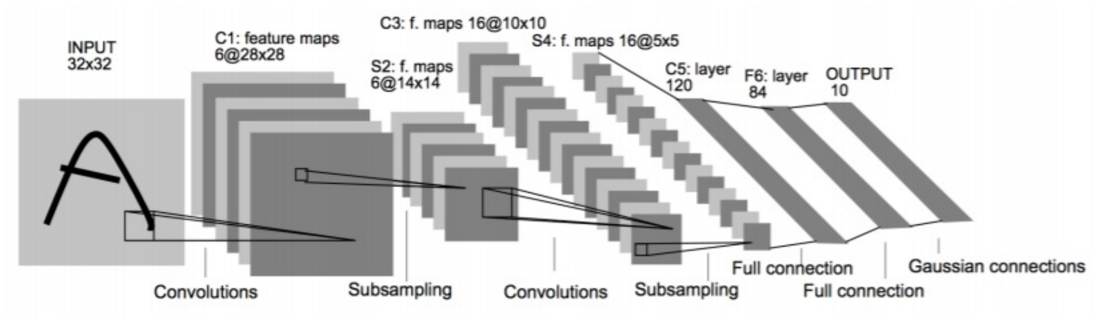


Figure 3.7.1: Architecture of LeNet-5 [19]

3.7.2 AlexNet

AlexNet, compared to LeNet, was deeper with 60 millions of parameters and bigger with 5 convolutional layers, 3 max-pooling, and 3 fully-connected layers (Figure 3.7.2). It popularized the ReLU as a non-linear function of choice [21]. AlexNet was submitted to the ImageNet ILSVRC challenge of 2012 and significantly outperformed the other handcrafted models (accuracy Top 5 of 84% compared to the second runner-up with 74%)

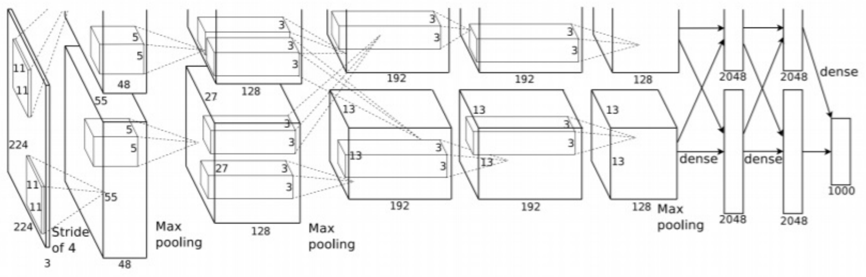


Figure 3.7.2: Architecture of Alexnet [21]

3.7.3 VGGNet

VGGNet, the runner-up architecture of ILSVRC2014 with almost 140 millions of parameters [22], has contributed to show that depth is a critical component for good performance. It uses much smaller 3×3 filters in each convolutional layers and combines them as a sequence of convolutions. The great advantage of VGGNet was the insight that multiple 3×3 convolutions in sequence can emulate the effect of larger receptive fields, for example 5×5 and 7×7 (Figure: 3.7.3). These ideas will be also used in more recent network architectures as Inception and ResNet.

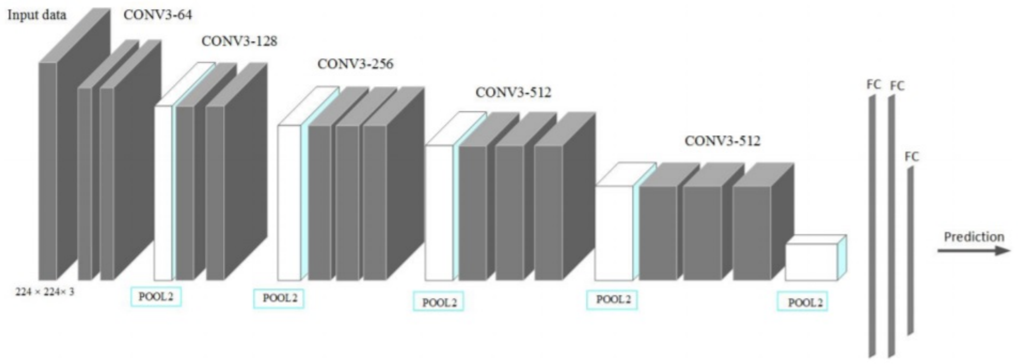


Figure 3.7.3: Architecture of VGGNet16 [23]

3.7.4 Inception

Inception, also known as GoogleNet, has dramatically reduced the number of parameters (40 million) [24]. At the top of the convolutional layers, it uses average pooling instead of fully connected layers, which helps it to eliminate a large number of parameters (Figure: 3.7.4). GoogleNet contains a stack of 22 convolutions and a new module called the Inception Module. This model is composed of parallel computations of convolutions of different shapes and max-pooling. The output from all these modules is concatenated at the end of each Inception module. The most recent architecture of Inception is InceptionV3 which uses batch normalization [26].

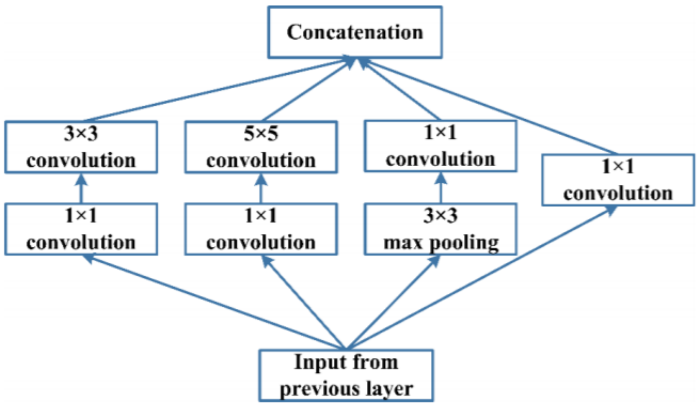


Figure 3.7.4: 1×1 convolutions are used to decrease the input size before 3×3 convolutions in order to provide more computational power such as in Inception [31]

3.7.5 ResNet

ResNet was the winner of ILSVRC 2015 with the top 5 error rate of only 3.6%, which is considered better than human-level accuracy [28]. Its main contribution was to use batch normalization and special skip connections for training deeper architectures (Figure: 3.7.5). ResNet with 1000 layers can be trained with those techniques. However, it has been empirically found that ResNet usually operates on blocks of relatively low depth (~ 20 – 30 layers), which act in parallel, rather than serially flow the entire length of the network [27].

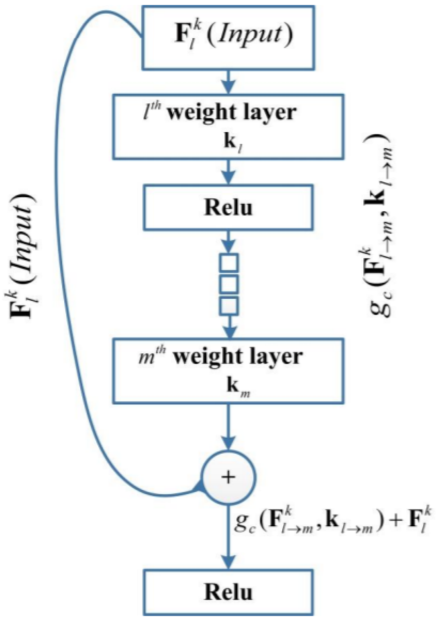

$\mathbf{F}_l^k$ (Input)
l^{th} weight layer $\mathbf{k}_l$
Relu
m^{th} weight layer $\mathbf{k}_m$
+
$g_c(\mathbf{F}_{l\to m}^k, \mathbf{k}_{l\to m}) + \mathbf{F}_l^k$
Relu
$\mathbf{F}_l^k$ (Input)
$g_c(\mathbf{F}_{l\to m}^k, \mathbf{k}_{l\to m})$

Figure 3.7.5: A skip connection is used to bypass the input to the next layers such as in ResNet [31]

3.7.6 MobileNet

MobileNets are based on a streamlined architecture that uses depth-wise separable convolutions to build lightweight deep neural networks [29]. The authors introduce two simple global hyperparameters that efficiently tradeoff between latency and accuracy. It is particularly useful for mobile and embedded vision applications. MobileNet V2 builds upon the concepts from MobileNet V1. It employs depth-wise separable convolution as efficient building blocks. However, MobileNet V2 introduces two new features to the architecture: 1) linear bottlenecks between layers, and 2) short connections between bottlenecks [30].

3.8 Recurrent Neural Network

In a traditional neural network, the assumption is that all inputs & outputs are independent. However, this is always not the case. For example, if we want to predict the word in a sentence, it is obvious that we have to know the words that came before it. Recurrent Neural Network (RNN) can map from the entire history of previous inputs to each output, whereas a traditional neural network can only map from input to output vectors. RNN can be seen as its having memory for storing unbounded history on previously processed elements. So at every point in time, this stored history is used to project the next output from the process (Figure: 3.8.1).

Long Short Term Memory network (LSTM) is a special kind of RNNs which is able to learn the long-term dependencies. It is explicitly designed to avoid the long-term dependency problem [32]. The LSTM cells allow the network to accumulate information, and once that information is exhausted, they can allow the network to forget the old state.

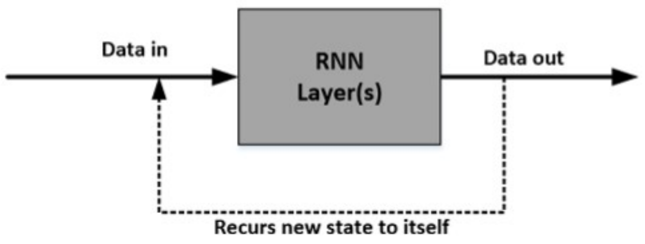


Figure 3.8.1: RNNs folded and unfolded state

3.9 Pre-trained Sentence Embeddings

Text data is typically raw and unstructured. These types of data need preprocessing before being fed to a neural network. One way is to use embedding, where the text data is processed by encoding words in a vector space model. This process involves adding a new dimension to this new vector for each feature. To be more specific, each unique word in the vocabulary can be considered as a feature and its presence will be marked by a unique integer. In the word embeddings, the initial approach is to learn word vector representations as part of the architecture of a language model. The model consists of an embedding layer, multiples intermediate layers, and one softmax layer. By learning distributed word representation and the probability of word sequences from these representations, the model predicts the next word in a sequence [34].

Pre-trained word embeddings can be created by training on a large text data set and used to classify a new dataset. However, in recent research, sentence embedding has shown promising results and stronger transferability between different tasks [35]. The Universal Sentence Encoder is introduced aiming at generating a fix-sized sentence encoding [33]. This sentence encoding model leverages the encoder from Transformer which is a multi-attention head that helps the model "attend" to the relevant information. Basically, the model computes multiple attention weighted sums, learning "context aware representations."

Chapter 4
Methodology

The chapter introduces our general framework for multi-modal fusion integrating visual and textual information from the cover.

4.1 Multi-modal Fusion with simple concatenation

Figure 4.1.1 describes our general framework for multi-modal fusion with simple concatenation. In this study, we refer to this as simple multi-model fusion. It takes the text embedding and image matrices as its inputs. We use a CNN architecture to extract the features from the image and a text classification architecture to extract the features from the text. We add two dense layers and concatenate these two layers with another dense layer. Then we use the sparse categorical-cross-entropy loss to train the neural network.

4.2 Multi-modal Fusion with DCCA

Now, we introduce deep canonical correlation analysis (DCCA) to book genre classification. DCCA was proposed in [38], and it computes representations of multiple modalities by passing them through multiple stacked layers of nonlinear transformations. Figure 4.2.1 depicts the structure of Multi-modal Fusion with DCCA used in this study. The details of DCCA is described as follows:

4.2.1 Canonical correlation analysis (CCA)

Canonical correlation analysis (CCA) is a widely used technique in the statistics community to measure the linear relationship between two multidimensional variables

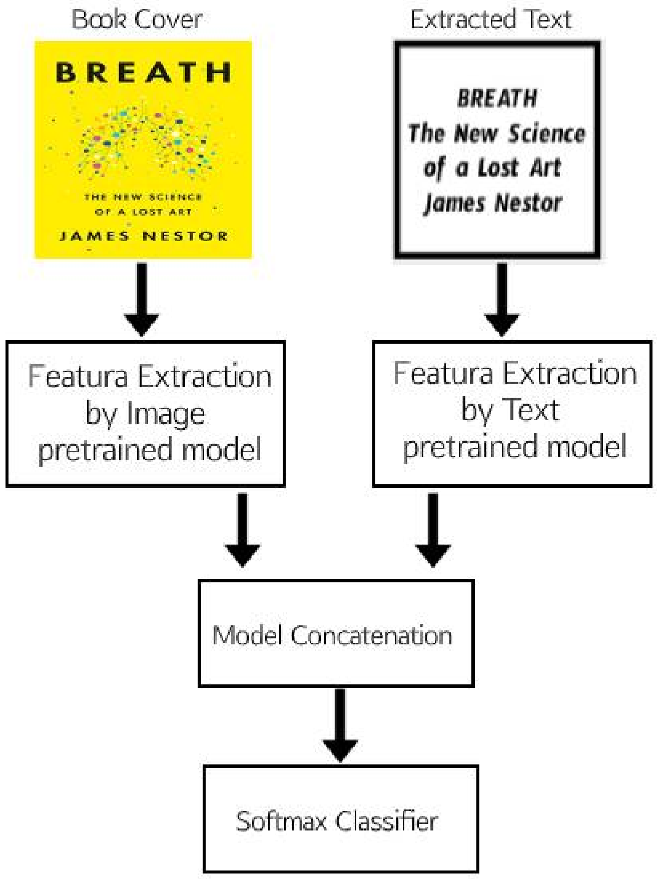


Figure 4.1.1: Simple multi-model fusion for the book cover classification

[36]. CCA was first applied to machine learning by Hardoon and colleagues [37]. It uses two views of a set of patterns and projects them onto a lower-dimensional space in which they are maximally correlated.

Given n pairs of centered samples $\{(x_i, y_i) \in \mathbb{R}^p \times \mathbb{R}^q\}_{i=1}^n$ with p and q as the sample dimensions, let $X = (x_1, x_2, x_3, ..., x_n) \in \mathbb{R}^{p\times n}$ and $Y = (y_1, y_2, y_3, ..., y_n) \in \mathbb{R}^{q\times n}$. CCA aims to seek pairs of projection directions $\alpha \in \mathbb{R}^p$ and $\beta \in \mathbb{R}^q$ such that the correlation coefficient of canonical projections $\alpha^T X$ and $\beta^T Y$ is maximized:

$$\rho(\alpha, \beta) = \frac{\alpha^T X Y^T \beta}{\sqrt{\alpha^T X X^T \alpha}\sqrt{\beta^T Y Y^T \beta}} = \frac{\alpha^T S_{xy} \beta}{\sqrt{\alpha^T S_{xx} \alpha}\sqrt{\beta^T S_{yy} \beta}} \tag{4.2.1}$$

where S_{xx} and S_{yy} are the covariance matrices of samples X and Y, and S_{xy} is the cross-covariance matrix matrix of X and Y. Obviously, the canonical correlation coefficient ρ is affine-invariant to arbitrary scaling of α and β. Therefore, CCA can be formulated as a constrained optimization problem:

$$(\alpha^*, \beta^*) = \arg\max_{\alpha, \beta} \alpha^T S_{xy} \beta \qquad (4.2.2)^1$$
$$\text{subject to} \quad \alpha^T S_{xx} \alpha = 1,\ \beta^T S_{yy} \beta = 1$$

To find multiple results of (α^i, β^i), subsequent projections are also constrained to be uncorrelated with previous projections, i.e., $\alpha^i S_{xx} \alpha^j = \beta^i S_{yy} \beta^j = 0$ for $i < j$. Combining the top k projection vectors α^i into a matrix $A \in \mathbb{R}^{p \times n}$ as column vectors and similarly placing β^i into $B \in \mathbb{R}^{q \times n}$, we then identify the top $k \leq min(p, q)$ projections:

$$\text{maximize:} \quad tr(A^T S_{xy} B) \tag{4.2.3}$$
$$\text{subject to} \quad A^T S_{xx} A = 1,\ B^T S_{yy} B = I$$

To solve this objective function, we first define $T = S_{xx}^{-\frac{1}{2}} S_{xy} S_{yy}^{-\frac{1}{2}}$, and we let U_k and V_k be the matrices of the first k left singular and right singular vectors of T, respectively. Then the optimal objective value is the sum of the top k singular values of T, and the optimum is obtained at $(A^*, B^*) = (S_{xx}^{-\frac{1}{2}} U_k, S_{yy}^{-\frac{1}{2}} V_k)$. More details about CCA

[1]The *argmax* of a function f defined on a set D as

$$\arg\max_{x \in D} f(x) = \{x | f(x) \geq f(y), \forall y \in D\}$$

solution can be found in [36]

4.2.2 Deep Canonical Correlation Analysis (DCCA)

Let, $X \in \mathbb{R}^{N \times p}$ is the extracted features matrix from the images and $Y \in \mathbb{R}^{N \times q}$ is the extracted features matrix from the texts. Here, N is the number of samples, and p and q are the dimensions of the extracted features for these two modalities, respectively. To transform the features of these two modalities, we use two deep neural networks as follows:

$$\begin{aligned} O_x &= f_x(X;\theta_x) \\ O_y &= f_y(Y;\theta_y) \end{aligned} \tag{4.2.4}$$

where, θ_x and θ_y are the parameters of the neural networks, $O_x \in \mathbb{R}^{N \times d}$ and $O_y \in \mathbb{R}^{N \times d}$ are the outputs, and d denotes the output dimension of the neural networks. The output O_x and O_y will be regularized by the Canonical Correlation Analysis(CCA). Our target is to jointly learn the parameters θ_x and θ_y for both neural networks such the correlation of O_x and O_y is as high as possible.

$$(\theta_x^*, \theta_y^*) = \underset{\theta_x, \theta_y}{\arg\max}\, corr(f_x(X;\theta_x), f_y(Y;\theta_y)) \tag{4.2.5}$$

To update θ_x and θ_y, we use the backpropagation algorithm described in Chapter 2. The solution for calculating the gradients of the objective function in Eq. 4.2.5 was developed in [38].

Let, $\bar{O_x}$ and $\bar{O_y}$ be the centered output matrices. Now, $\bar{O_x} = O_x^T - \frac{1}{N} O_x^T \mathbf{1}$ and $\bar{O_y} = O_y^T - \frac{1}{N} O_y^T \mathbf{1}$. We define $\hat{S_{xy}} = \frac{1}{N-1} \bar{O_x} \bar{O_y}^T$, $\hat{S_{xx}} = \frac{1}{N-1} \bar{O_x} \bar{O_x}^T + r_x$ and $\hat{S_{yy}} = \frac{1}{N-1} \bar{O_y} \bar{O_y}^T + r_y$. Here, r_x and r_y are regularization constants. The total correlation of the top k components of O_x and O_y is the sum of the top k singular values of matrix

$T = \hat{S_{xx}}^{-\frac{1}{2}} \hat{S_{xy}} \hat{S_{yy}}^{-\frac{1}{2}}$. In this study, we take $k = d$, and the total correlation is the trace of T:

$$corr(O_x, O_y) = \big(tr(T^T T)\big) \tag{4.2.6}$$

The derivation of the gradient is given in the appendix of [38]. If the singular value decomposition of T is $T = UDV^T$, then

$$\frac{\partial corr(O_x, O_y)}{\partial O_x} = \frac{1}{N-1}(2\nabla_{xx}\bar{O_x} + \nabla_{xy}\bar{O_y}), \tag{4.2.7}$$

where,

$$\begin{aligned} \nabla_{xx} &= -\frac{1}{2}\hat{S_{xx}}^{-\frac{1}{2}} UDU^T \hat{S_{xx}}^{-\frac{1}{2}} \\ \nabla_{xy} &= \hat{S_{xx}}^{-\frac{1}{2}} UV^T \hat{S_{yy}}^{-\frac{1}{2}} \end{aligned}$$

and $\frac{\partial corr(O_x, O_y)}{\partial O_y}$ has a symmetric expression.

To make a fusion between O_x and O_y, we merge those together as follows:

$$O = \begin{bmatrix} O_x \\ O_y \end{bmatrix} \tag{4.2.8}$$

The final output O will be used to train an Support Vector Machine(SVM) classifier to determine the book genre [42].

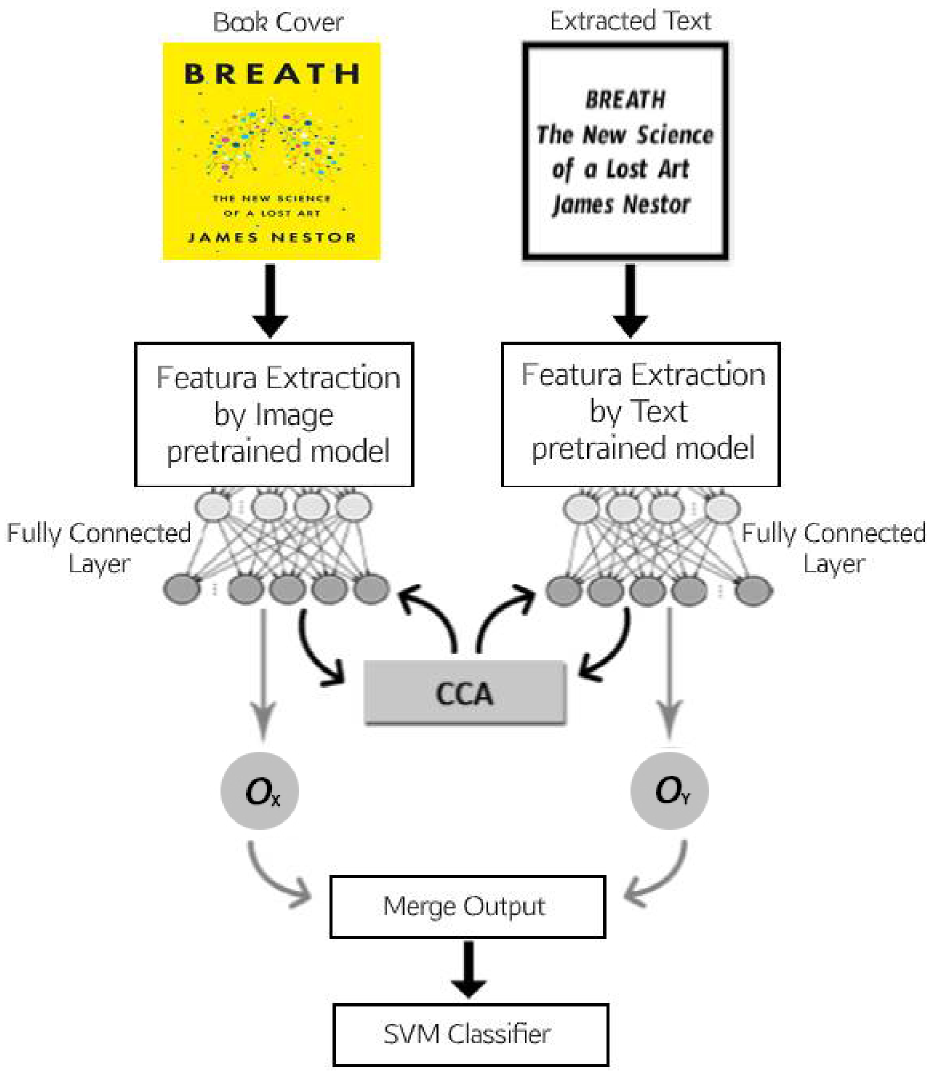


Figure 4.2.1: The structure of the proposed DCCA for the book cover classification

Chapter 5
Experiments

This chapter describes experiments carried out to classify books based on cover. These included comparisons among the baseline methods and text-image based method we propose here. We also discuss evaluation of the impacts of different activation functions, and visualization of learned features.

5.1 Dataset

To evaluate the proposed methodology, we use the BookCover30 dataset proposed by Iwana and Uchida 2016 [5]. This dataset contains 57,000 book cover images divided into 30 genres (Table 5.1.1).

Arts & Photography	Biographies & Memoirs
Business & Money	Calendars
Children's Books	Christian Books & Bibles
Comics & Graphic Novels	Computers & Technology
Cookbooks, Food & Wine	Crafts, Hobbies & Home
Engineering & Transportation	Health, Fitness & Dieting
History	Humor & Entertainment
Law	Literature & Fiction
Medical Books	Mystery, Thriller & Suspense
Parenting & Relationships	Politics & Social Sciences
Reference	Religion & Spirituality
Romance	Science & Math
Science Fiction & Fantasy	Self-Help
Sports & Outdoors	Teen & Young Adult
Test Preparation	Travel

Table 5.1.1: The 30 genres of the BookCover30 dataset

Each genre has 1900 samples. We divide the data set into three parts: 80% training samples, 10% validation samples and 10% test samples.

We use Google Cloud Vision API[1] to extract the texts from the cover images. To use these texts in our classification task, we need to encode our text using a process called tokenization that we split each text into a list of words or tokens and assign a unique integer value to all the words. However, we encode only those words that appeared at least 5 times in the whole dataset. It helps us remove the insignificant words for classification.

For the cover images, no pruning and no class membership corrections were done. In addition, to overcome the memory limitation, we resized all of the images to 224px by 224px.

5.2 Evaluation

As baseline methods, we evaluate different pretrained models such as VGGNet-16, MobileNet-V1, MobileNet-V2, ResNet50, and Inception-V2 only on the cover images. These models are pretrained on the ImageNet dataset. We added a flatten layer, a fully connected layer of 1024 hidden units, and a fully connected layer of 30 units on top of every pretrained model. For the texts, we use a simple vanilla LSTM with 256 memory units, one dimensional Convolutional network with 256 filters, and stride 5 and pretrained Universal Sentence Encoder model. For LSTM, we use word embedding and for Universal Sentence Encoder, we feed the unprocessed texts. We output all the models with softmax activation. For every model, we use sparse-categorical-cross-entropy as the loss function with Adam optimizer with the default learning rate(0.001).

In order to measure the performance of each of the models, we need a good performance metric. Since all of the categories in the dataset have the same number

[1]https://cloud.google.com/vision/docs/ocr

of samples, accuracy serves as a good metric. We compare these standalone models using accuracy defined as

$$Accuracy = \frac{\text{No. of correct}}{\text{No. of total}} \times 100\%$$

We compare the Top 1 and Top 3 accuracy which means that the correct genre gets to be in the Top 1 and Top 3 probabilities respectively for it to count as accurate. All the experiments are conducted using Python language in Google Colab[2] using an Nvidia Tesla K80 GPU. It will use Keras[3], an open-source deep learning library, with the TensorFlow[4] backend, a free machine learning platform.

After evaluating the baseline models, we choose the best model for image and the best model for text that we use in the multi-modal fusion. We find that the best model for cover image classification is ResNet50 which has an accuracy of around 29.56% and for text, Universal Sentence Encoder with an accuracy of 53.58%. For the simple multi-modal fusion, we freeze these pretrained models and extract the features using the ReLU activation function defined in **chapter 3.2**. Then we concatenate these features and feed into a final fully connected layer with a softmax activation. We use RMSprop to optimize the sparse-categorical-cross-entropy loss for this model.

Finally, for the proposed multi-modal fusion with DCCA, we use the same structure as we use in multi-modal fusion, except the CCA loss as the loss function. Through CCA, we jointly learn the weight and bias parameters of the fully connected layers and the final output then is used to train an SVM classifier.

[2]https://colab.research.google.com/
[3]https://keras.io/
[4]https://www.tensorflow.org/

5.3 Results

The comparison of accuracy for Top 1 and Top 3 categories among the models is shown in **Table 5.3.1**. We also include the accuracy of LeNet and AlexNet presented on [5]. ResNet50 gets more than 6% and 21% gains in accuracy in comparison to that of AlexNet for Top 1 and Top 3 respectively. The accuracy after adding the text modality is achieved around 100% gain compared to any image-based model. Despite the great improvement over the image modality, we get only 4% gain in the concatenated model over the Universal Sentence Encoder.

	Model	Accuracy	
		Top 1	Top 3
Only Image	LeNet [5]	13.5	24.7
	AlexNet [5]	27.8	40.3
	MobileNet V1	27.22	46.40
	MobileNet V2	23.6	42.03
	ResNet50	**29.56**	**49.02**
	Inception V2	26.21	45.61
Only Text	LSTM(256)	41.49	61.59
	Universal Sentece Encoder	**52.58**	**73.70**
Image & Text	**Simple Multi-modal Fusion**	**54.69**	**73.70**
	Multi-modal Fusion with DCCA	**48.93**	**72.28**

Table 5.3.1: Accuracy comparison of book genre classification

Table 5.3.2 shows genre accuracy comparison between simple multi-modal fusion and multi-modal fusion with DCCA. We find that the best accuracy(89% and 83% respectively) is achieved in the *Cookbooks, Food & Wine* category and the *Teen & Young Adult* category has the worst performance of 12% and 18% respectively. Overall, simple multi-modal fusion has the best result among all the models we experiment with.

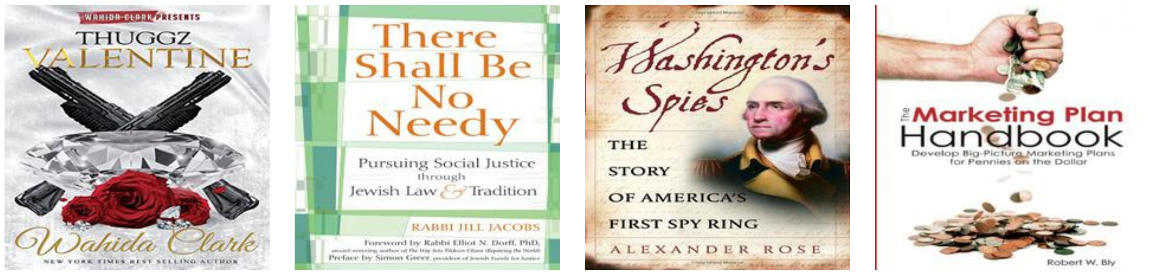
Genre	Accuracy	
	Simple Multi-model Fusion	Multi-model Fusion with DCCA
Arts & Photography	48	39
Biographies & Memoirs	33	33
Business & Money	58	49
Calendars	86	81
Children's Books	39	52
Comics & Graphic Novels	66	66
Computers & Technology	78	69
Cookbooks, Food & Wine	89	83
Crafts, Hobbies & Home	68	46
Christian Books & Bibles	67	54
Engineering & Transportation	63	50
Health, Fitness & Dieting	52	40
History	53	44
Humor & Entertainment	32	22
Law	70	47
Literature & Fiction	27	28
Medical Books	73	56
Mystery, Thriller & Suspense	67	61
Parenting & Relationships	49	49
Politics & Social Sciences	33	22
Reference	41	35
Religion & Spirituality	51	44
Romance	72	66
Science & Math	63	40
Science Fiction & Fantasy	46	48
Self-Help	55	45
Sports & Outdoors	63	49
Teen & Young Adult	12	18
Test Preparation	79	77
Travel	56	51

Table 5.3.2: Genre Accuracy Comparison

5.4 Analysis

Book genre classification based on cover images utilizing deep learning has already been approached in some studies [5, 11, 10]. In this study, we also made some efforts to classify books based on cover information. Despite these efforts, none of the above-mentioned approaches has achieved notable improvements in terms of accuracy. So a natural question is whether the task is doable leveraging the current deep learning methods.

To answer the question, we find a major issue in the dataset. The dataset was collected from the book cover images and genres listed by Amazon.com as the top categories under "Books". In the source data, many of the books belonged to multiple categories, however, during the prepossessing, one category at random for those multi-label books has been selected in [5]. For example, in **Figure 5.4.1(b)**, the book belonged to Law and Religion & Spirituality both, however, it is mainly a Law book, but touches upon Religion & Spirituality. The latter category is chosen randomly. This random selection affected negatively to the prediction.


WAHIDA CLARK PRESENTS
THUGGZ VALENTINE
Wahida Clark
NEW YORK TIMES BEST SELLING AUTHOR
There Shall Be No Needy
Pursuing Social Justice through Jewish Law & Tradition
RABBI JILL JACOBS
Washington's Spies
THE STORY OF AMERICA'S FIRST SPY RING
ALEXANDER ROSE
The Marketing Plan Handbook
Develop Big-Picture Marketing Plans for Pennies on the Dollar
Robert W. Bly

(a) **(b)** **(c)** **(d)**

Figure 5.4.1: Miscategorized covers. (a) Predicted: Romance; Original: Literature & Fiction, (b) Predicted: Law; Original: Religion & Spirituality, (c) Predicted: History; Original: Biographies & Memoirs (d) Predicted: Business & Money; Original: Self-Help

To better understand the dataset, we analyze the confusion matrix (**Figure 5.4.2**) that we get from Simple Multi-modal Fusion. The *Teen & Young Adult* rarely occurs exclusively and is mixed with the other genres like the *Romance*, the *Sports & Outdoors*, the *Mystery, Thriller & Suspense*, etc. This is understandable because the books in the *Romance*, the *Sports & Outdoors*, the *Mystery, Thriller & Suspense* are often written on targeting young people.

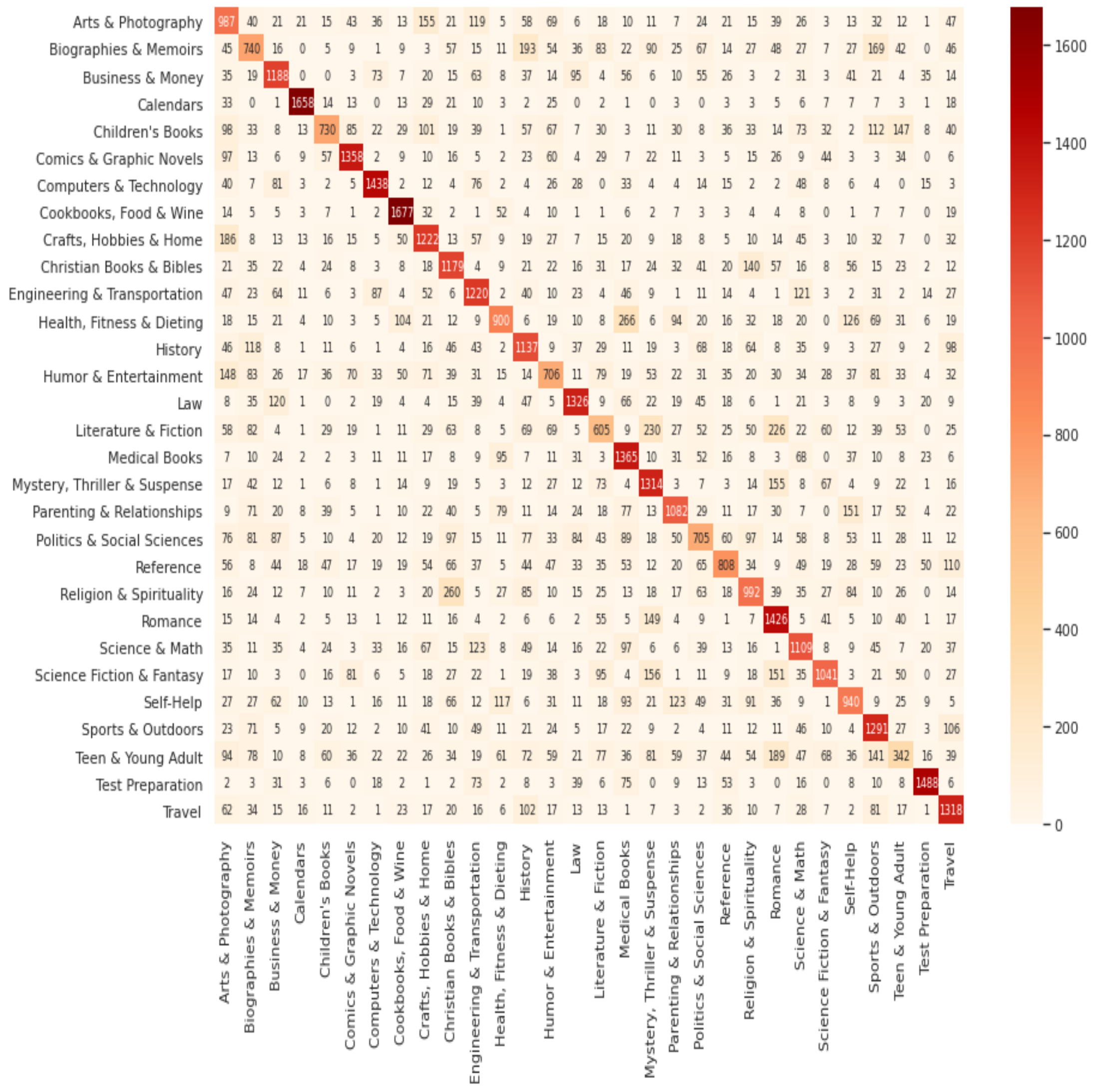
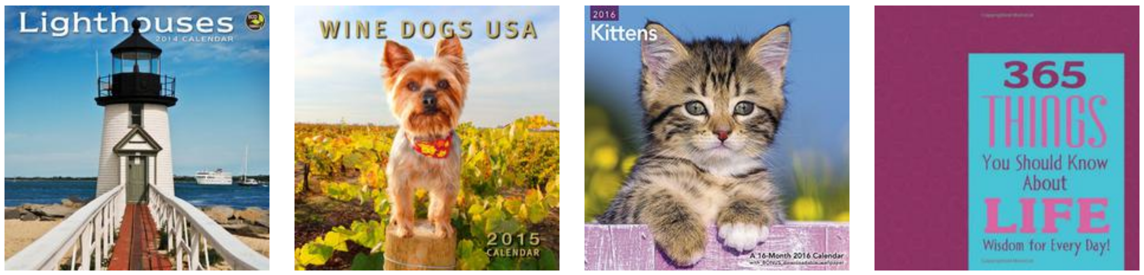


Figure 5.4.2: Confusion matrix for Multi-modal fusion.

There is an overlap between the *Christian Books & Bibles* books and the *Religion & Spirituality* books. This is obvious, because the *Christian Books & Bibles* category is rather a sub-category of the *Religion & Spirituality* category. It is likely the Amazon.com addresses the American customers by creating whole category for the *Christian Books & Bibles* books.

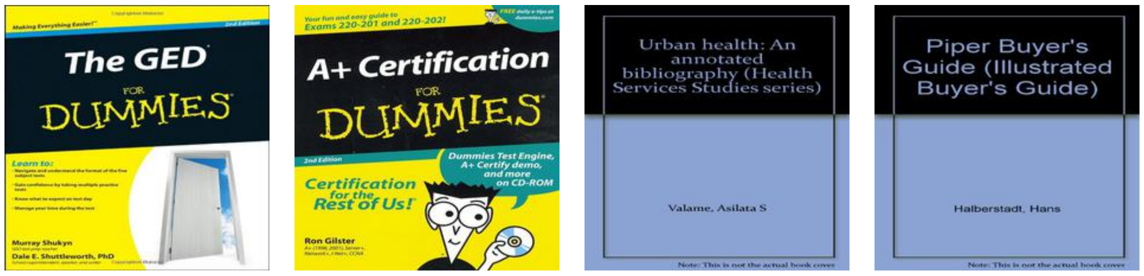


(a) **(b)** **(c)** **(d)**

Figure 5.4.3: (a), (b) and (c) is predicted Calendar and (d) is predicted as Self-help. The correct label for all these books is Calendar

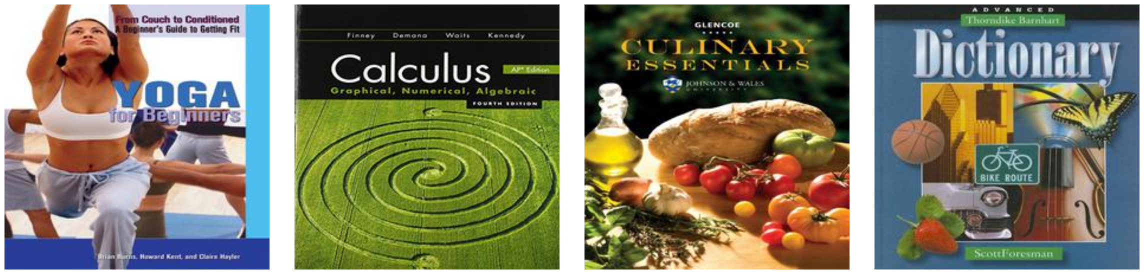
The most accurately categorized class is the *Cookbooks, Food & Wine* category, though, there are a few overlaps with the *Crafts, Hobbies & Home* and the *Health, Fitness & Dieting*. The second most accurate category predicted by Multi-modal Fusion is *Calendars*. From **Figure 5.4.3**, (a), (b) and (c) contains the words 'calendar' and are correctly categorized as Calendar, however, (d) looks different than the others, hence, is misclassified as Self-help.

We also found that the dataset has low inter-class variance and high intra-class variance, which makes it extremely difficult for the models method to deal with. Low inter-class variance pertains to the fact that the book covers belonging to different categories are strikingly similar. In **Figure 5.4.4**, (a) & (b) and (c) & (d) look alike but belong to different categories. In contrast, high intra-class variance relates to the fact that there is a huge variety of different looking covers present in a single category. For instance, all the covers in **Figure 5.4.5** belong to the *Teen & Young*


The GED for DUMMIES
A+ Certification for DUMMIES
Urban health: An annotated bibliography (Health Services Studies series)
Valame, Asilata S
Piper Buyer's Guide (Illustrated Buyer's Guide)
Halberstadt, Hans

(a) (b) (c) (d)

Figure 5.4.4: Low inter-class variance example.



(a) (b) (c) (d)

Figure 5.4.5: High intra-class variance example.

Adult, however, look very different from each other. We can see that the embedding space is still highly overlapping, highlighting the complexity of the problem. There are also categories that seem to be well segregated such as the Test Preparation class since the covers are highly distinctive in that case.

Figure 5.4.6 visualizes the the softmax activations obtained from Multi-modal Fusion. The T-SNE plot is highly overlapping at the center, which clearly portrays the complexity of the problem. The *Calendars* and the *Cookbooks, Food & Wine* are well separated due to their uniqueness from other categories.

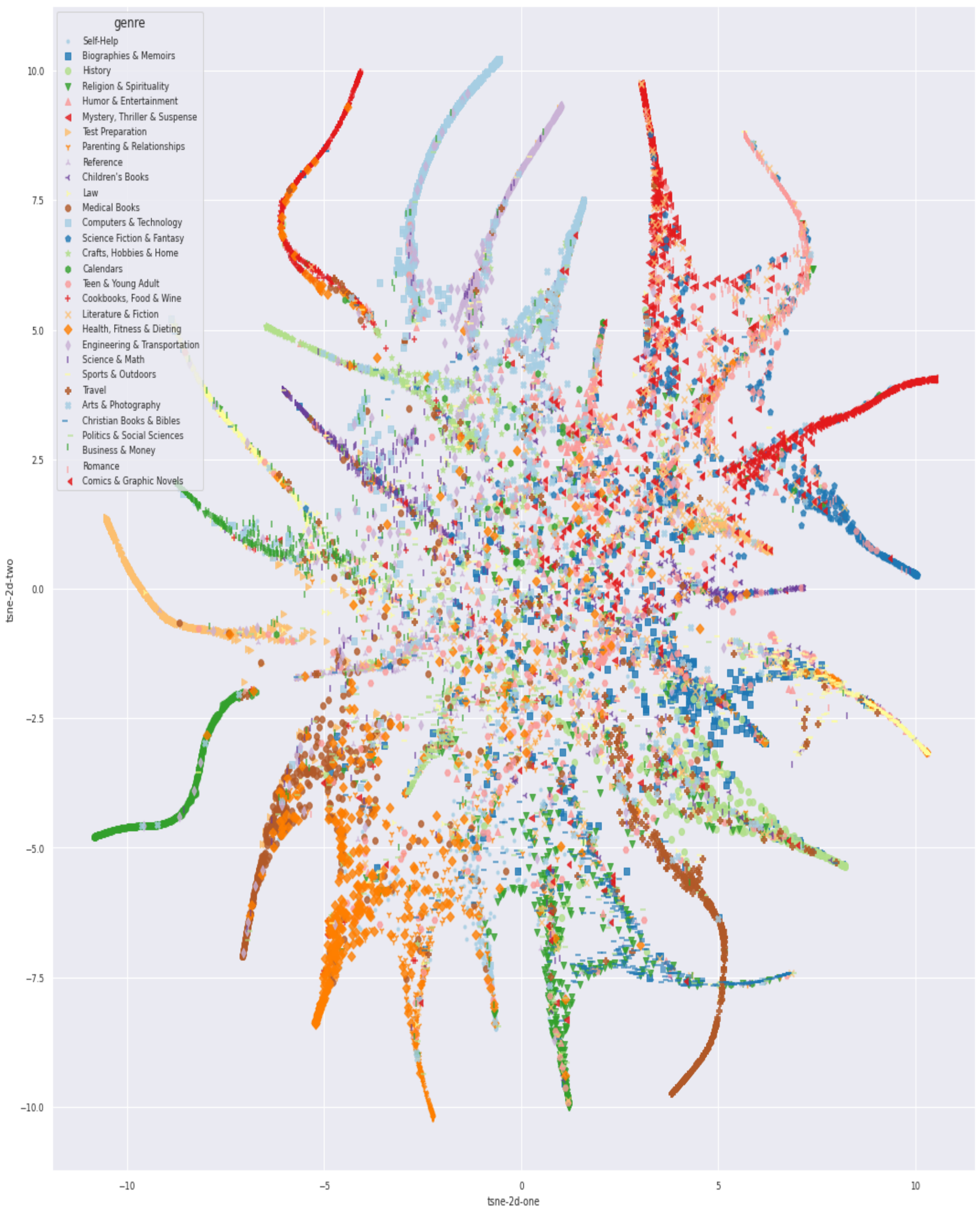


Figure 5.4.6: T-SNE plot of BookCover30 dataset using softmax activations, obtained from Multi-modal Fusion.

Chapter 6
Conclusion and Future Work

In this study, we employed different state-of-the-art deep learning models to classify books based on its cover. We have presented two multi-view learning approaches: the simple multi-modal fusion and the multi-modal fusion with DCCA. In our experiment, the simple multi-modal fusion had a Top 1 accuracy of 54.69% and outperforms all the other models. The accuracy of the multi-modal fusion with DCCA is 48.93% which is less than that of the simple multi-modal fusion. A reasonable explanation of this performance is the lack of connection between the visual information and the textual information for some of the books. The weakly labeled covers can be another reason for this poor performance.

In the future, an improved classification performance can be obtained by cleaning the mislabeled data or turning this problem into a multi-label classification task. Incorporating more information such as effect of coloration, font style, illustration style, etc. can also enhance accuracy.

Bibliography

[1] Karayev, Sergey, Matthew Trentacoste, Helen Han, Aseem Agarwala, Trevor Darrell, Aaron Hertzmann, and Holger Winnemoeller. ”Recognizing image style.” arXiv preprint arXiv:1311.3715 (2013).

[2] Ivasic-Kos, Marina, Miran Pobar, and Luka Mikec. ”Movie posters classification into genres based on low-level features.” In 2014 37th international convention on information and communication technology, electronics and microelectronics (MIPRO), pp. 1198-1203. IEEE, 2014.

[3] Ivasic-Kos, Marina, and Miran Pobar. ”Multi-label Classification of Movie Posters into Genres with Rakel Ensemble Method.” In International Conference on Innovative Techniques and Applications of Artificial Intelligence, pp. 370-383. Springer, Cham, 2017.

[4] Chu, Wei-Ta, 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, pp. 39-45. 2017.

[5] Iwana, Brian Kenji, Syed Tahseen Raza Rizvi, Sheraz Ahmed, Andreas Dengel, and Seiichi Uchida. ”Judging a Book by its Cover.” arXiv preprint arXiv:1610.09204 (2016).

[6] Gozuacik, Necip, and C. Okan Sakar. ”Turkish Movie Genre Classification from Poster Images using Convolutional Neural Networks.” In 2019 11th International Conference on Electrical and Electronics Engineering (ELECO), pp. 930-934. IEEE, 2019.

[7] Cetinic, Eva, Tomislav Lipic, and Sonja Grgic. ”Fine-tuning convolutional neural networks for fine art classification.” Expert Systems with Applications 114 (2018): 107-118.

[8] Petrovski, Petar, and Anna Lisa Gentile. ”Can you judge a music album by its cover?.” In (KNOW@ LOD/CoDeS)@ ESWC. 2016.

[9] Oramas, Sergio, Oriol Nieto, Francesco Barbieri, and Xavier Serra. ”Multi-label music genre classification from audio, text, and images using deep features.” arXiv preprint arXiv:1707.04916 (2017).

[10] Kjartansson, Sigtryggur, and Alexander Ashavsky. ”Can you Judge a Book by its Cover?.” (2017). http://cs231n.stanford.edu/reports/2017/pdfs/814.pdf

[11] Buczkowski, Przemyslaw, Antoni Sobkowicz, and Marek Kozlowski. ”Deep Learning Approaches towards Book Covers Classification.” In ICPRAM, pp. 309-316. 2018.

[12] Cetinic, Eva, and Sonja Grgic. ”Genre classification of paintings.” In 2016 International Symposium ELMAR, pp. 201-204. IEEE, 2016.

[13] Christopher M Bishop et al. Pattern recognition and machine learning, volume 1. Springer New York, 2006.

[14] Andrej Krenker, J Bester, and Andrej Kos. Introduction to the artificial neural networks. Artificial neural networks: methodological advances and biomedical applications. InTech, Rijeka. ISBN, pages 978–953, 2011.

[15] Leshno, Moshe, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. ”Multilayer feedforward networks with a nonpolynomial activation function can approximate any function.” Neural networks 6, no. 6 (1993): 861-867.

[16] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In Neural Networks: Tricks of the Trade, pages 437–478. Springer, 2012

[17] Ruder, Sebastian. "An overview of gradient descent optimization algorithms." arXiv preprint arXiv:1609.04747 (2016).

[18] Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting." The journal of machine learning research 15, no. 1 (2014): 1929-1958.

[19] LeCun, Yann, et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86.11(1998): 2278-2324.

[20] LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. Nature 521, 436–444 (2015). https://doi.org/10.1038/nature14539

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012

[22] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[23] Muhammad, Usman, Weiqiang Wang, Shahbaz Pervaiz Chattha, and Sajid Ali. "Pre-trained VGGNet Architecture for Remote-Sensing Image Scene Classification." In 2018 24th International Conference on Pattern Recognition (ICPR), pp. 1622-1627. IEEE, 2018.

[24] Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. ”Going deeper with convolutions. arXiv 2014.” arXiv preprint arXiv:1409.4842 1409 (2014).

[25] Iandola, Forrest N., Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. ”SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and¡ 0.5 MB model size.” arXiv preprint arXiv:1602.07360 (2016).

[26] Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. ”Rethinking the inception architecture for computer vision.” In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2818-2826. 2016.

[27] Veit, Andreas, Michael J. Wilber, and Serge Belongie. ”Residual networks behave like ensembles of relatively shallow networks.” In Advances in neural information processing systems, pp. 550-558. 2016.

[28] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. ”Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.” In Proceedings of the IEEE international conference on computer vision, pp. 1026-1034. 2015.

[29] Howard, Andrew G., Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. ”Mobilenets: Efficient convolutional neural networks for mobile vision applications.” arXiv preprint arXiv:1704.04861 (2017).

[30] Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. ”Mobilenetv2: Inverted residuals and linear bottlenecks.” In Pro-

ceedings of the IEEE conference on computer vision and pattern recognition, pp. 4510-4520. 2018.

[31] Khan, Asifullah, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. ”A survey of the recent architectures of deep convolutional neural networks.” arXiv preprint arXiv:1901.06032 (2019).

[32] Hochreiter, Sepp, and Jürgen Schmidhuber. ”Long short-term memory.” Neural computation 9, no. 8 (1997): 1735-1780.

[33] Cer, Daniel, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant et al. ”Universal sentence encoder.” arXiv preprint arXiv:1803.11175 (2018).

[34] Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. ”A neural probabilistic language model.” Journal of machine learning research 3, no. Feb (2003): 1137-1155.

[35] Conneau, Alexis, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. ”Supervised learning of universal sentence representations from natural language inference data.” arXiv preprint arXiv:1705.02364 (2017).

[36] Hotelling, Harold. ”Relations between two sets of variates.” In Breakthroughs in statistics, pp. 162-190. Springer, New York, NY, 1992.

[37] Hardoon, David R., Sandor Szedmak, and John Shawe-Taylor. ”Canonical correlation analysis: An overview with application to learning methods.” Neural computation 16, no. 12 (2004): 2639-2664.

[38] Andrew, Galen, Raman Arora, Jeff Bilmes, and Karen Livescu. "Deep canonical correlation analysis." In International conference on machine learning, pp. 1247-1255. 2013.

[39] Bassiou, Nikoletta, Constantine Kotropoulos, and Anastasios Papazoglou-Chalikias. "Greek folk music classification into two genres using lyrics and audio via canonical correlation analysis." In 2015 9th international symposium on image and signal processing and analysis (ISPA), pp. 238-243. IEEE, 2015.

[40] Qiu, Jie-Lin, Wei Liu, and Bao-Liang Lu. "Multi-view emotion recognition using deep canonical correlation analysis." In International Conference on Neural Information Processing, pp. 221-231. Springer, Cham, 2018.

[41] Liu, Wei, Jie-Lin Qiu, Wei-Long Zheng, and Bao-Liang Lu. "Multimodal Emotion Recognition Using Deep Canonical Correlation Analysis." arXiv preprint arXiv:1908.05349 (2019).

[42] Vapnik, Vladimir. "The support vector method of function estimation." In Nonlinear Modeling, pp. 55-85. Springer, Boston, MA, 1998.