electronics

MDPI

Article

Enhancing the Accuracy of an Image Classification Model Using Cross-Modality Transfer Learning

Jiaqi Liu, Kwok Tai Chui * and Lap-Kei Lee *

Department of Electronic Engineering and Computer Science, School of Science and Technology, Hong Kong Metropolitan University, Hong Kong, China; s1304012@live.hkmu.edu.hk
* Correspondence: jktchui@hkmu.edu.hk (K.T.C.); lklee@hkmu.edu.hk (L.-K.L.)

Abstract: Applying deep learning (DL) algorithms for image classification tasks becomes more challenging with insufficient training data. Transfer learning (TL) has been proposed to address these problems. In theory, TL requires only a small amount of knowledge to be transferred to the target task, but traditional transfer learning often requires the presence of the same or similar features in the source and target domains. Cross-modality transfer learning (CMTL) solves this problem by learning knowledge in a source domain completely different from the target domain, often using a source domain with a large amount of data, which helps the model learn more features. Most existing research on CMTL has focused on image-to-image transfer. In this paper, the CMTL problem is formulated from the text domain to the image domain. Our study started by training two separately pre-trained models in the text and image domains to obtain the network structure. The knowledge of the two pre-trained models was transferred via CMTL to obtain a new hybrid model (combining the BERT and BEiT models). Next, GridSearchCV and 5-fold cross-validation were used to identify the most suitable combination of hyperparameters (batch size and learning rate) and optimizers (SGDM and ADAM) for our model. To evaluate their impact, 48 two-tuple hyperparameters and two well-known optimizers were used. The performance evaluation metrics were validation accuracy, F1-score, precision, and recall. The ablation study confirms that the hybrid model enhanced accuracy by 12.8% compared with the original BEiT model. In addition, the results show that these two hyperparameters can significantly impact model performance.

Keywords: batch size; cross-modality; deep learning; image classification; learning rate; overfitting; text classification; transfer learning

Citation: Liu, J.; Chui, K.T.; Lee, L.-K. Enhancing the Accuracy of an Image Classification Model Using Cross-Modality Transfer Learning. *Electronics* **2023**, *12*, 3316. https://doi.org/10.3390/electronics12153316

Academic Editor: Hyunjin Park

Received: 30 June 2023
Revised: 29 July 2023
Accepted: 1 August 2023
Published: 2 August 2023

1. Introduction

Image classification problems have been leading research in computer vision. With the continual development of the Internet in recent decades, people can easily create, access, and analyze all types of images, which has resulted in the rapid expansion of the number of images. Images are an important way of carrying information and are essential in all aspects of people's daily communication, life, and work. In this context, there has been an emphasis on finding accurate and valuable images in a short amount of time from many images. The potential of machine learning algorithms (particularly deep learning algorithms) is increasingly being explored as technology advances, and it has produced beneficial effects in various sectors, including, but not limited to, natural language processing (NLP), traffic prediction, medical diagnosis, and image classification [1]. Attention is drawn to image classification problems because of their state-of-the-art performance in the field. However, machine learning must improve with lengthy training times, the large sample sizes required, and limited computer ability [2].

With the advent of deep learning algorithms, automatic feature extraction from images can be achieved. Convolutional neural networks (CNNs) [3] are one of the most mainstream image analysis methods [4]. Regarding deep learning models, it is desirable to have

sufficient labeled training data to achieve promising model performance (e.g., accurate and unbiased classification). However, some real-world problems are linked to small-scale labeled datasets, such as rare diseases [5], mental health [6], and legal areas [7]. Transfer learning has recently been suggested as a solution to this issue, which has several benefits for enhancing the performance of target models from single or multiple source models [8,9]. The general idea of transfer learning is to transfer knowledge learned from the source domain to the target domain, speeding up training and lowering the requirement for sample size in the target dataset. Some studies have demonstrated the improvement of transfer learning on image classification accuracy and the effect of transfer learning on CNN, which performs better in image classification after pre-training compared to traditional CNN [10,11]. In the methodologies of [12–14], including another domain as the source domain becomes redundant if the training samples are large enough and an impressive performance can be achieved while restricted in the target domain. There are various levels of disagreement between different source and target domain data pairs. Regardless of their disagreement, imposing knowledge from the source domain into the target domain can lead to some performance degradation or, in worse cases, disrupt data consistency in the target domain [15]. On the other hand, traditional transfer learning is only partially applicable to some tasks and requires a good degree of similarity or common information between the source and target domains. As mentioned above, the key part of the transfer learning algorithm is to discover the similarity between the source domain $P_S(X, Y)$ and the target domain $P_T(X, Y)$. When the labeled target data are not available $(n_l = 0)$, one has to resort to the similarity between the marginal $P_S(X)$ and $P_T(X)$; although this does have a theoretical limitation [14]. In contrast, this problem can be solved if a significant number of samples $(x_l, y_l) \sim P_T(X, Y)$ and $(x_s, y_s) \sim P_s(X, Y)$ are available. Thus, a reasonable migration learning algorithm may be able to use datasets with labeled target domains to mitigate the negative impact of irrelevant source information [16]. In other words, transferring learning between domains with low similarity will be prone to negative transfer [16–18], i.e., resulting in degradation of the performance of the target model.

Such a problem of transfer learning between domains with low similarity is known as cross-modality transfer learning, which involves transfer learning between heterogeneous datasets [19]. In this paper, a breakthrough is desired to alleviate the limits of traditional transfer learning when the source and target domains differ. A cross-modality transfer approach from text to images is chosen. It is believed that the machine learning methods used for text classification could be used for image classification, known as cross-modality transfer.

1.1. Related Work on Cross-Modality Transfer Learning

The discussion of existing works includes only research studies using cross-modality transfer learning, i.e., existing works using traditional transfer learning with high similarity between the source and target domains are not considered. Therefore, cross-modality transfer learning was proposed to tackle the issue of negative transfer between heterogeneous source and target domains [20–25].

Image to Image. Lei et al. [20] performed cross-modality transfer learning using ResNet-50 with three convolutional layers from ImageNet (the source dataset) to the ICPR2012 dataset or the ICPR2016 dataset (the target datasets). The ratio between the training and testing datasets was 80:20. The model achieved an accuracy of 97.1% (an improvement of 6.12%) for the ICPR 2012 dataset and an accuracy of 98.4% (an improvement of 0.163%) for the ICPR 2016 dataset. In another work [21], knowledge was transferred from the NPHEp-2 dataset (source dataset) to the LSHEp-2 dataset (target dataset) using a parallel deep residual network with a two-dimensional discrete wavelet transform. The training-testing dataset was in an 80:20 ratio. The proposed method enhanced the accuracy by 0.417% (from 95.9% to 96.3%). Hadad et al. [22] proposed using cross-modality transfer learning to improve the recognition rate of masses in breast MRI images. They trained a network on X-ray images and then transferred the pre-trained network to the target

domain (MRI images). Performance evaluation revealed that cross-modality transfer learning improved the classification performance from an overall accuracy of 90% to 93%. Their study's limitation is that it involves transferring between different types of images, specifically from X-ray images to MRI images. While X-ray images have a relatively small dataset compared to other domains (e.g., the text domain), the transfer process still fails to fully utilize the benefits of CMTL due to the relatively large amount of data in MRI images. Another work [23] proposed a cross-modality transfer learning approach from 2D to 3D sensors in which different modalities shared the same observation targets. They employed a pre-trained model network based on 2D images and then transferred the pre-trained model to the visual system of 3D sensors. The model achieved an average precision improvement of 13.2% and 16.1% compared to ConvNets and ViTs, respectively. A cross-modality transfer learning algorithm was proposed for transferring a network trained on a large dataset in the source domain (RGB) to the target domains (depth and infrared) [24], which was used for the task of transferring knowledge from one source modality to another target modality without accessing task-related source data. The model achieved an accuracy of 90.2% in the single-source cross-modality knowledge transfer task from RGB to NIR using the RGB-NIR dataset without task-related source data and 92.7% from NIR to RGB. However, their designed model has yet to be tested in tasks with larger modality gaps as it was only applied in cases with smaller modality differences.

Text to Image. Du et al. [25] described a chest X-Ray quality assessment method that combined image-text contrastive learning and medical domain knowledge fusion. The proposed method integrated large-scale real clinical chest X-rays and diagnostic report text information and fine-tuned the pretrained model based on contrastive text-image pairs. The model yielded an accuracy of 89.7–97.2% for 13 classes. Another work [26] proposed a zero-shot transfer learning model that can recognize objects in images without any training samples available. The model acquired knowledge by learning from an unsupervised, large-scale text corpus. In the performance evaluation, the images were split into visible and invisible categories. The model achieved about 80% accuracy in the training categories. The research study also suggested that if two zero-shots had no remote similarity with any visible class, the performance was relatively poor, resulting in suboptimal zero-shot classification. Chen et al. [27] presented a history-aware multimodal transformer (HAMT) approach for visual linguistic navigation (VLN). The HAMT encoded all past panoramic observations by a hierarchical visual transformer, which can effectively incorporate far-future history into multimodal decision-making. The model joins text, history, and current observations to predict the following actions. Another work [28] compared pre-trained and fine-tuned representations at the visual, verbal, and multimodal levels using a set of detection tasks and introduced a new dataset specifically for multimodal detection. While their visual-linguistic models could understand color at the multimodal level, they relied on biases in the textual data concerning object position and size. This suggests that fine-tuning the visual-linguistic model in a multimodal task does not necessarily improve its multimodal capabilities. In [29], a new efficient and flexible multimodal fusion method called prompt-based multimodal fusion (PMF) was proposed that utilized a unimodal pre-trained transformer. The authors presented a modular multimodal fusion framework that enabled bidirectional interactions between different modalities to dynamically learn different objectives of multimodal learning. The proposed method is memory-efficient, which can significantly reduce the use of training memory and achieve comparable performance to existing fine-tuning methods with fewer trainable parameters. However, the performance of PMF on all three datasets still lags behind the baseline tuning with the same pre-trained backbone and no tuning of hyperparameters. In addition, CLiMB consisted of several implementations of CL algorithms and an improved visual language translator (ViLT) model that could be deployed on both multimodal and unimodal tasks [30]. It was found that common language learning methods could help mitigate forgetting in multimodal task learning but did not enable cross-task knowledge transfer.

Other. Falco et al. [31] collected a visual dataset and a tactile dataset to form the nature of the distant source and target domains. Cross-modality transfer learning was supported by subspace alignment and transfer component analysis for dimensionality reduction and a geodesic flow kernel for characterizing geodesic flow. The model achieved an accuracy of 89.7%. A multimodal transformer framework with variable-length memory (MTVM) was proposed for VLN [32]. The framework also included an explicit memory bank for storing past activations. It enabled the agent to easily update the temporal context by adding the current output activation corresponding to the action at each step to learn a strong relationship between the instruction and the temporal context, thus further improving navigation performance.

1.2. Research Limitations of Existing Works

By analyzing existing research papers, we can identify their limitations. Most current research involves similar domains, such as cross-domain studies within the Image-to-Image field. In the Text-to-Image field, good performance can be achieved by making an ideal model if the data in the source and target domains are similar [25]. However, considering the zero-shot transfer learning problem [26], when the data in the source and target domains are dissimilar or have low similarity, the performance of the target model is poor, which illustrates that the current research in the Text-to-Image field is still limited by the similarity between the source and target domains. In other fields, such as the previously mentioned research from the visual to the tactile domain, the performance is good, with high accuracy. However, the applicability is limited, making it suitable for niche areas but not widely applicable.

1.3. Our Research Contributions

Cross-modality transfer learning is considered for text-to-image classification problems. First, we adopt bidirectional encoder representations from the transformer (BERT) model, typically trained in two stages [33]. The first stage uses MaskLM to train the language model, mask a random portion of words in a sentence, and predict the masked words by understanding the context. In the second stage, the BERT model predicts the following sentence, which helps it better understand the relationship between individual sentences. We used BERT to train text sentiment classification on the IMDb reviews dataset, which contains 25,000 movie reviews for training and 25,000 movie reviews for testing, explicitly used for sentiment classification. In addition, we employ a bidirectional encoder representation from the image transformer (BEiT) model [34]. This self-supervised learning model applies a similar idea to the BERT model to the image classification task. The idea is to obtain image features by masking the image modeling pre-training task, achieving an accuracy of 83.2 in the ImageNet-1K classification task, which we used to train on the ImageNet-1K dataset for image classification. Finally, a novel hybrid model is designed by joining the first ten layers of the pre-trained BERT model and the last two layers of the pre-trained BEiT model. An ablation study showed that the contribution of the BEiT model enhanced accuracy by 12.8%.

Regarding the performance evaluation of the hybrid model, we have conducted an in-depth analysis of the model's performance with the batch size, learning rate, and types of optimizers.

1.4. Organization of the Paper

The rest of the paper is organized as follows: Section 2 introduces the datasets and illustrates the methodology of the novel hybrid model. Then, a performance evaluation of the proposed model is conducted, comparing the proposed work with existing work. To study the contributions of the standalone BERT model and the standalone BEiT model, an ablation study is carried out in Section 4. Finally, a conclusion is drawn, and research implications and future research directions are discussed.

2. Materials and Methods

In this section, all stages of cross-modality transfer learning are illustrated. First, two datasets are used to train the models in two different domains, i.e., the image and text domains, and save the training results as the pre-trained models for the next stage. In the second stage, we combined the two pre-trained models and selected CIFAR-10 as the dataset for the next stage of training. In the third stage, to obtain the most suitable optimizer, batch size, and learning rate for the model, we used both GridSearchCV and K-Fold cross-validation methods. The performance is evaluated using different hyperparameters and optimizers by calculating the F1-score, precision, and recall. The whole process of cross-modal transfer learning will be summarized. Following the workflow, the BERT and BEiT models are first pretrained using the IMDb reviews and ImageNet-1K datasets, respectively. Then, the knowledge is transferred to the novel hybrid model. Afterward, the CIFAR-10 dataset is pre-processed to determine whether the 5-fold cross-validation has been completed. If it is not yet complete, the combination of optimizers and hyperparameters is fed into the unique hybrid model, and if it is, training and testing are finished using the best optimizer and hyperparameters. In the 5-fold cross-validation process, the dataset is first divided into five parts, with one part selected as the testing data and four parts as the training data for each training session. Each set of hyperparameters is cross-validated five times, and the mean result is calculated. The results were then compared to select the best combination of hyperparameters. In normal model training, we calculated the results without averaging them.

2.1. Pre-Training Models

The main objective of this section is to use the pre-trained model as a feature extractor by pre-training the model on a large dataset. We first trained the model on a large underlying dataset; in the text domain, we chose to use the BERT model on the IMDb review dataset, a widely used sentiment binary classification dataset, as a benchmark for sentiment classification, which consists of 100,000 text reviews of films. Half (50,000) of the reviews contained no labels, and these were used for testing, with the other 50,000 reviews paired with labels of 0 or 1, representing negative and positive sentiment, respectively. These reviews with tags were split into two groups, with each group having 12,500 positive and 12,500 negative reviews to keep the data balanced. These labels are linearly mapped from IMDb's star rating system, in which critics can rate a film with a certain number of stars from 1 to 10 [35]. Figure 1 shows the split of the IMDb review dataset and two examples of reviews. The BERT model is a pre-trained model proposed by the Google AI Institute that has demonstrated impressive performance in all aspects, using a network architecture with a multi-layer transformer structure, which is most distinctive in that it does not use traditional recurrent neural networks (RNNs) and CNNs; instead, it uses an attention mechanism to convert the distance between two arbitrarily placed positions. This solves the problem of long-term dependency in NLP. It has already achieved wide application in the field of NLP.

	Positive	Negative
Training	12,500	12,500
Validation	12,500	12,500

Positive Example:
Someone release this movie on DVD so it can take its hallowed place as on of the greatest films of all time in ten to twenty years when critics and film historians look back on the so-called films of the 1990's and see how vapid they were for the most part, and how Lars Von Trier tried to revolutionize and revitalize the international film world with this masterpiece.

Negative Example:
I have this film out of the library right now and I haven't finished watching it. It is so bad I am in disbelief. Audrey Hepburn had totally lost her talent by then, although she'd pretty much finished with it in 'Robin and Marian.' This is the worst thing about this appallingly stupid film. It's really only of interest because it was her last feature film and because of the Dorothy Stratten appearance just prior to her homicide.

Figure 1. The split of the IMDb review dataset and two examples of reviews.

In the image domain, we chose to use the BEiT model for training on the ImageNet-1K dataset, which is currently the largest image recognition dataset in the world and is

mainly used in machine vision, target detection, and image classification. The ImageNet-1K dataset introduced for the ILSVRC 2012 visual recognition challenge has been at the center of modern advances in deep learning. ImageNet-1K is the primary dataset for pre-training computer vision migration learning models, and improving the performance of ImageNet-1K is often seen as a litmus test for general applicability to downstream tasks. ImageNet-1K is a subset of the full ImageNet dataset, which consists of 14197122 images divided into 21841 classes. We will refer to the full dataset as ImageNet-21K, and ImageNet-1K was created by selecting a subset of 1.2 million images belonging to 1000 mutually exclusive classes from ImageNet-21K [36]. In contrast, the BEiT model is a self-supervised visual representation model proposed by Microsoft, which is similar to BERT in that it uses the transformer's masked image modeling task. Specifically, in pre-training, each image has two views. The developer converts the original image into a tokenizer, then randomly masks some patches and feeds them into the transformer. Experimental results in image classification and semantic segmentation show that the BEiT model achieves better results. Figure 2 shows the whole process of pre-training the BERT and BEiT models. The BERT model was trained using the IMDb Reviews dataset as an input, whereas the BEiT model was trained using the ImageNet-1K dataset. Their weights and network structures after pre-training are saved, and some of them (knowledge) will be transferred to a novel hybrid model in a later step, which is known as knowledge transfer. The selection of the number of layers from the pre-trained BERT and BEiT models will be elaborated in Section 2.2. The left half of Figure 3 illustrates the pre-training process for BERT and BEiT, with BERT being pre-trained in the IMDb reviews dataset and BEiT being pre-trained in ImageNet-1K.

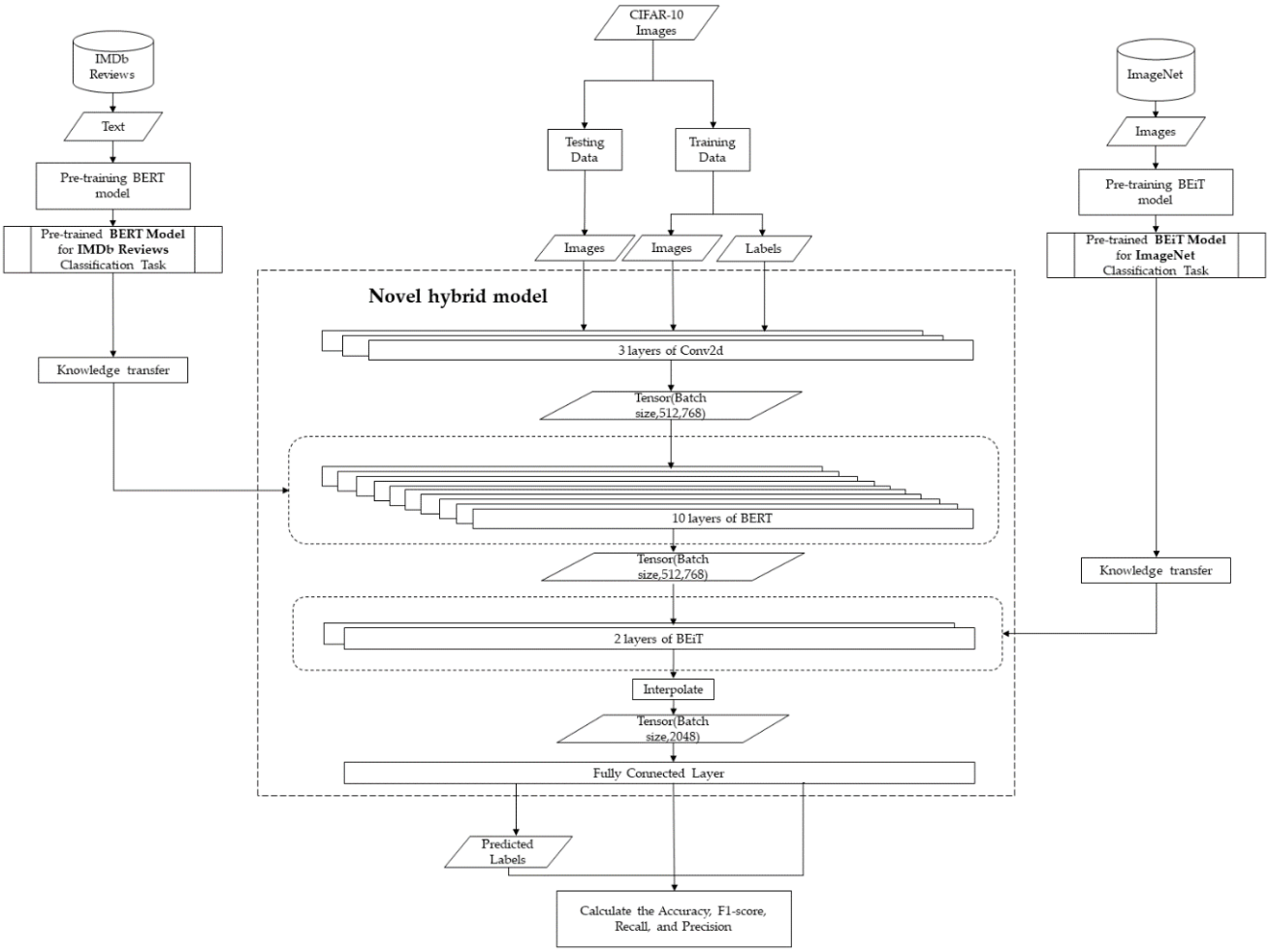


Figure 2. The process of pre-training the BERT and BEiT models.

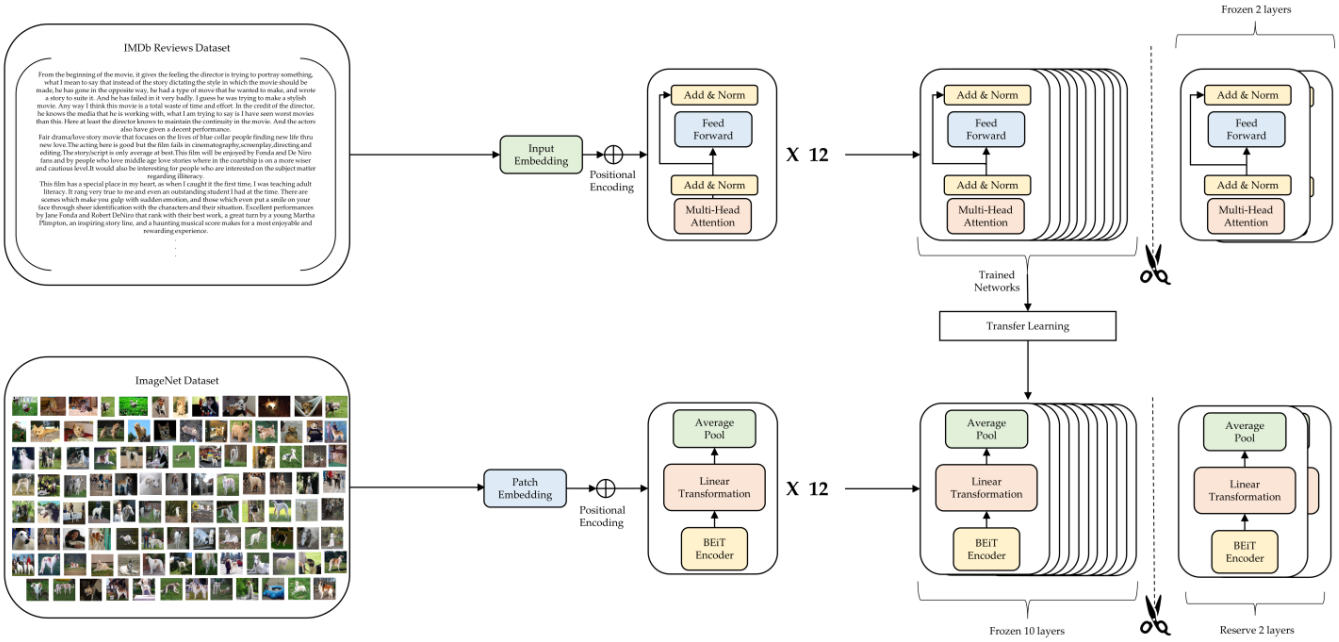


Figure 3. The process of pre-training and transfer learning for BERT and BEiT and the structure of the new hybrid model.

2.2. Design of a Novel Hybrid Model

To achieve cross-modal transfer learning, we combined the BERT and BEiT models. By merging the two models, we can transfer a large amount of knowledge learned by the BERT model in the source domain to the task in the target domain to compensate for the lack of data in the target domain. The first ten layers of the BERT model and the last two layers of the BEiT model are retained. The last few layers of a neural network are usually specialized; Yosinski et al.'s study [37] claims that the last layer allows features to transition from general to specific with some specificity. In contrast, the first few layers are usually not specific to a particular dataset or task but generic as they apply to many datasets and tasks; therefore, we chose to retain the last two layers of BEiT, which would make the novel hybrid model better suited to image classification tasks. The other layers are frozen and are not used for training. Liu et al. [38] showed that the transformer-based structure is more transferable to other tasks in the middle layer, while the higher layers are more task-specific. Kirichenko et al. [39] demonstrated that the retraining of the last layer improves the performance of the model and improves its robustness. This suggests that the results are heavily influenced by the last linear layer of the model and that even though the model has acquired the features of the data in the previous layers, the last layer can still assign higher weights to the data. Kovaleva et al.'s study [40] calculated the similarity between pre-trained and fine-tuned BERT weights by finding that the weights of the last two layers changed the most after fine-tuning. This suggests that the last two layers of the BERT model learn the most information in a given task and that the previous layers mainly capture more underlying base information. Based on these studies, we believe that removing the last two layers of BERT can help the new hybrid model better learn the basics of BERT while retaining the specificity of the BEiT model for better classification tasks. Then, we add the corresponding network structures and weights of the pre-trained BERT and BEiT models to a new hybrid model for the next stage of training. Cross-modality transfer learning is used to extract information features from the pre-trained datasets, which could be used to extract deep features from new images. Therefore, these models may help accomplish image classification tasks. Our novel hybrid model processes the input image through 3 convolutional layers and the ReLU activation function; then, the processed image is considered a tensor with shape (batch size, 512, 768); next, this tensor is passed into the first ten layers of the BERT encoder, and the output tensor is passed as an input to the

BEiT model; then, using the interpolation method, the output tensor is resized to (batch size, 2048) using interpolation; the elements of the first dimension are extracted; finally, these elements are passed to the fully connected layers; the final output with shape (batch size, 10) is obtained through the fully connected layers. The right half of Figure 3 shows the transfer learning process of the two pre-trained models and the structure of the new hybrid model, where the knowledge of the first ten layers of BERT is transferred to the new model. In contrast, the first ten layers of BEiT are frozen, keeping the last two layers for the image classification task. Table A1 (Appendix A) explains the detailed structure of our new hybrid model, including the layer's type, output shape, and parameters, and concludes with a summary of the model's parameters and sizes.

2.3. GridSearchCV and K-Fold Cross-Validation

To find the best combination of batch size and learning rate for the new hybrid model, the traditional GridSearchCV method is used to find the best hyperparameters. In this process, the CIFAR-10 dataset is trained using 48 combinations of BS (4, 8, 12, 16, 20, 24, 28, and 32), LR (0.005, 0.001, 0.0005, 0.0001, 0.00005, and 0.00001), optimizers (stochastic gradient descent with momentum (SGDM), and adaptive moment estimation (ADAM)). Because of the momentum involved, SGDM is faster than SGD, training will be faster than SGD, and local minima can be an escape to achieve global minima. Simply put, momentum enables SGD to locate the global minima more quickly and precisely. Both SGDM and ADAM are two of the most popular optimizers. In typical applications, the ADAM optimizer takes advantage of faster initial learning, whereas the SGDM optimizer yields a more accurate model in the later stage. It can be explained by the fact that the ADAM optimizer has added the adaptive learning rate mechanism on top of the SGDM optimizer, which enables the ADAM optimizer to increase the optimization speed by assigning different learning rates for different parameters. Being an adaptive learning rate algorithm, ADAM determines unique learning rates for various parameters. RMSprop and stochastic gradient descent with momentum can be combined to form ADAM. Similar to RMSprop, it scales the learning rate using gradient squaring, and like SGDM, it leverages momentum by utilizing a moving average of the gradient rather than the gradient itself. Figure 4 illustrates this process and all combinations of the hyperparameters used in the 5-fold cross-validation. Figure 5 illustrates the CIFAR-10 dataset with 5-fold cross-validation and training in our novel hybrid model.

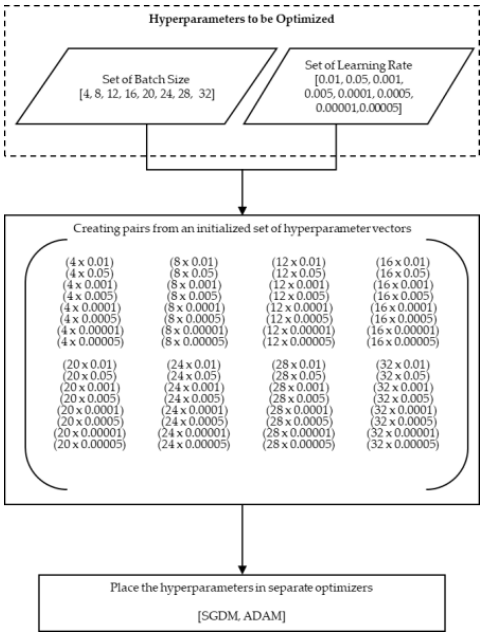


Figure 4. All combinations of hyperparameters were used in the 5-fold cross-validation.

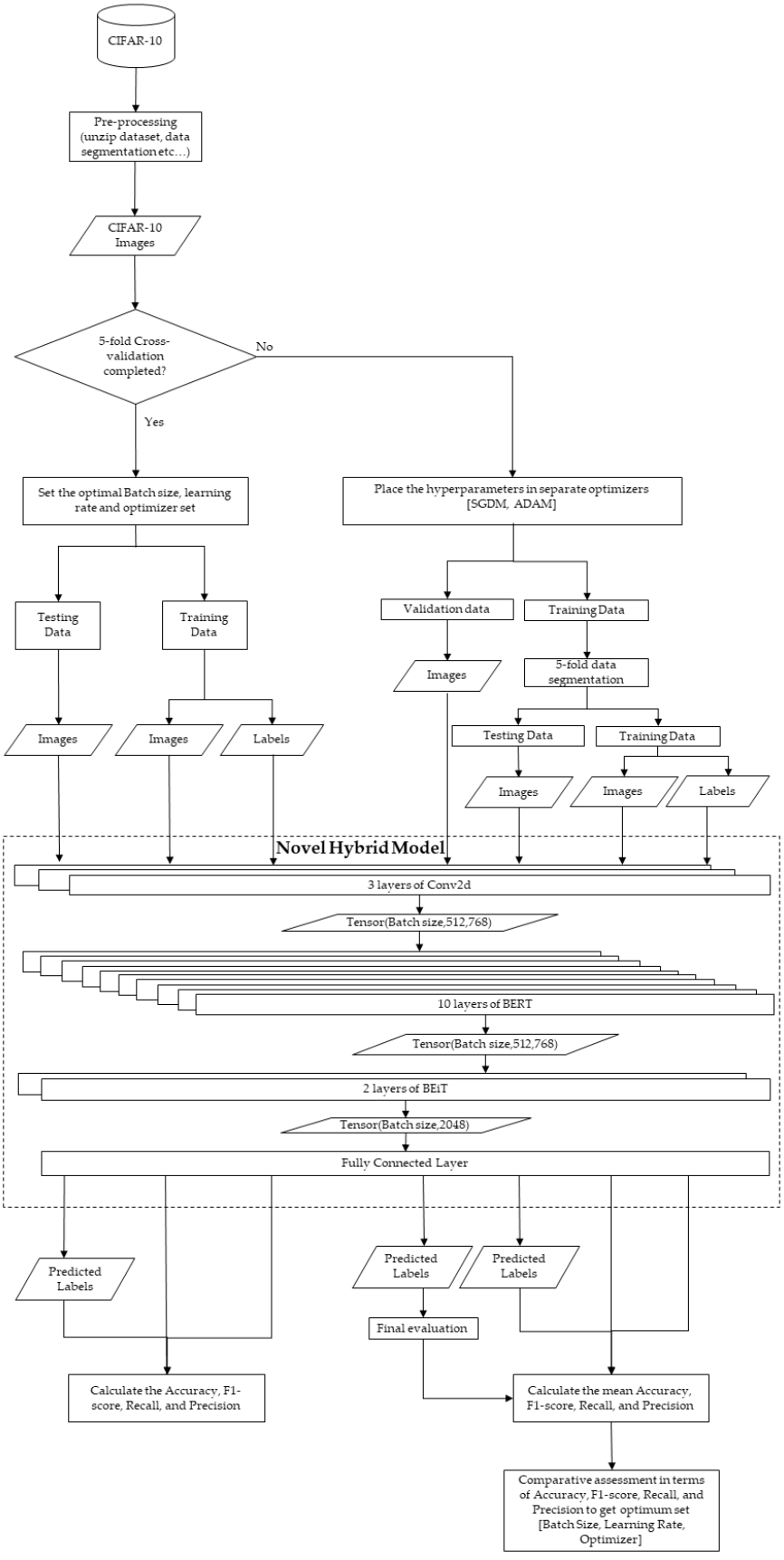


Figure 5. CIFAR-10 dataset with 5-fold cross-validation and training in novel hybrid model.

The performance of this hybrid model is then evaluated using K-fold cross-validation with K = 5 [41], which divides the dataset into K groups, with each subset of data serving as a separate validation set and the remaining K-1 subset of data serving as the training set. Each fold takes 10 epochs to complete. The reason for this design is that we found in the training of our previous hybrid model that the model was usually overfitted at around 10 calendar hours. The validation set results are evaluated separately, and the final mean squared error (MSE) is summed and averaged to obtain the cross-validation error. Figure 6 shows the process of 5-fold cross-validation. Cross-validation efficiently uses the limited data available, and the evaluation results are as close to the model's performance on the test set as possible. Unique values for the optimal hyperparameters batch size and learning rate were determined by comparing the F1-score (Equation (1)), precision (Equation (2)), and recall (Equation (3)) of each set of hyperparameters after K-fold cross-validation [42]. When the hybrid model is used to classify the CIFAR-10 dataset, we obtain the optimal hyperparameter values (BS = 24 and LR = 0.0005) for the SGDM optimizer, which results in an F1-score of 57.79%, a precision of 59.6481%, and a recall of 61.6944%.

$$F1 - score = 2 \times \frac{Recall \times Precision}{Recall + Precision} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

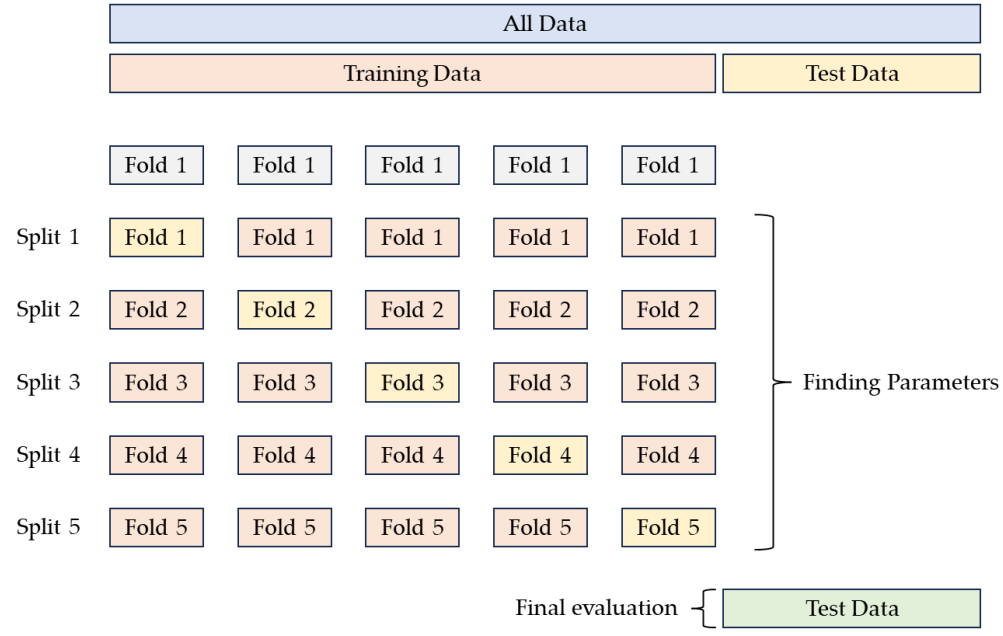


Figure 6. Process of 5-fold cross-validation.

3. Experimental Setup and Results Analysis

The experimental setup is based on the methodology described in Section 2. All simulations are conducted using a PC with NVIDIA GEFORCE GTX 3090—24 GB Graphics, a 15 vCPU AMD EPYC 7543 32-Core Processor, and Python 3.8.

3.1. 5-Fold Cross-Validation

Regarding 5-fold cross-validation, the dataset was divided into 80% and 20% for training and testing of the model, respectively. To evaluate and validate the impact of both hyperparameters, we increased the number of samples in the specified ranges of

the LR (Equation (4)) and BS (Equation (5)) to obtain a detailed output distribution for better interpretation. In determining the range of LR, we found that both optimizers were prone to non-convergence when they used LRs greater than 0.005, so the maximum LR is set at 0.005. For other specific values of LR, they refer to Usmani et al.'s research [43] to finalize the range of LR. For the range of BS, we chose the most common from 4 to 32, with BS increasing by eight at a time. This study used an extended Cartesian product matrix consisting of 48 two-tuple hyperparameters generated from the following two vectors:

$$LR\epsilon[0.005,\ 0.001,\ 0.0005,\ 0.0001,\ 0.00005,\ 0.0001] \tag{4}$$

$$BS\epsilon[4,\ 8,12,\ 16,\ 20,\ 24,\ 28,\ 32] \tag{5}$$

In addition, the model is evaluated using SGDM and ADAM. Table 1 summarizes the performance of each set of hyperparameters, including the average of all parameters and standard deviation of validation accuracy for 5-fold at each cross-validation. The summarized parameters are used in addition to the validation accuracy, and we use three measures: F1-score, recall, and precision.

Table 1. The performance of each set of hyperparameters.

BS	LR	Optimizer	Standard Deviations (%)	Validation Accuracy (%)	F1-Score (%)	Recall (%)	Precision (%)
4	0.00001	SGDM	1.5727	56.32	32.889	34.667	33.833
		ADAM	3.0496	59.18	32.333	33.999	31.500
	0.00005	SGDM	0.9613	61.31	41.667	43.333	40.833
		ADAM	0.2296	9.97	4.666	11.666	2.916
	0.0001	SGDM	1.9611	59.15	36.333	37.333	36.833
		ADAM	0.2872	9.91	3.775	14.167	2.239
	0.0005	SGDM	24.8728	40.49	34.444	38.333	32.500
		ADAM	0.3000	9.95	4.000	10.000	2.500
	0.001	SGDM	0.1523	9.95	6.444	11.667	4.583
		ADAM	0.3968	9.95	6.444	11.667	4.583
	0.005	SGDM	0.2340	10.00	4.444	6.667	3.333
		ADAM	0.2028	10.12	2.000	5.000	1.250
8	0.00001	SGDM	1.0712	51.43	40.444	41.845	43.155
		ADAM	0.3269	57.49	47.500	50.575	50.238
	0.00005	SGDM	0.5180	44.00	43.996	46.607	47.698
		ADAM	0.3088	9.96	2.222	10.000	1.250
	0.0001	SGDM	3.5695	60.03	44.978	48.714	47.000
		ADAM	0.3309	10.05	2.222	10.000	1.250
	0.0005	SGDM	18.3248	45.78	42.306	45.250	44.806
		ADAM	0.2555	10.15	3.704	14.000	2.167
	0.001	SGDM	21.3882	26.95	13.534	23.048	11.869
		ADAM	0.2740	9.96	3.111	14.000	1.750
	0.005	SGDM	0.2279	10.04	2.182	4.000	1.500
		ADAM	0.2098	10.03	3.111	14.000	1.750
12	0.00001	SGDM	0.8184	48.61	35.667	36.167	36.667
		ADAM	0.9671	57.77	53.299	55.867	53.001
	0.00005	SGDM	1.3184	59.23	42.667	42.000	44.000
		ADAM	24.6269	29.94	12.667	12.000	14.000
	0.0001	SGDM	1.5544	60.93	40.899	42.107	42.024
		ADAM	0.2972	9.99	4.444	6.667	3.333

Table 1. *Cont.*

BS	LR	Optimizer	Standard Deviations (%)	Validation Accuracy (%)	F1-Score (%)	Recall (%)	Precision (%)
12	0.0005	SGDM	0.7146	59.27	37.133	35.833	40.333
		ADAM	0.1212	9.93	10.444	21.667	7.083
	0.001	SGDM	0.1469	10.01	4.667	11.667	2.917
		ADAM	0.1937	10.09	4.667	11.667	2.917
	0.005	SGDM	0.1754	10.06	6.444	11.667	4.583
		ADAM	0.2196	10.02	2.000	5.000	1.250
16	0.00001	SGDM	1.3853	44.46	28.779	32.283	29.200
		ADAM	0.5009	56.74	52.315	56.839	53.749
	0.00005	SGDM	1.5716	58.73	42.184	47.870	43.685
		ADAM	24.3304	39.23	36.593	42.256	37.370
	0.0001	SGDM	1.9416	58.47	51.295	54.241	55.635
		ADAM	0.2595	9.87	3.257	11.944	1.910
	0.0005	SGDM	2.1745	60.06	50.436	53.204	53.153
		ADAM	0.0963	9.88	2.795	11.944	1.667
	0.001	SGDM	24.2741	39.84	31.150	35.167	32.365
		ADAM	0.2001	10.11	1.373	9.444	0.747
	0.005	SGDM	0.1659	9.92	1.373	7.222	0.764
		ADAM	0.0508	9.87	1.987	9.413	0.908
20	0.00001	SGDM	0.9337	41.47	32.568	36.577	34.750
		ADAM	1.1632	57.38	47.456	50.374	50.921
	0.00005	SGDM	1.2411	58.17	49.420	52.798	55.075
		ADAM	20.2989	50.38	43.375	48.237	44.785
	0.0001	SGDM	1.2448	57.15	49.800	55.042	50.093
		ADAM	0.2402	10.08	1.070	6.944	0.583
	0.0005	SGDM	1.9982	59.45	47.849	49.890	51.778
		ADAM	0.0837	9.76	1.575	9.722	0.861
	0.001	SGDM	19.3078	48.44	38.313	40.950	41.583
		ADAM	0.0989	10.11	2.334	11.944	1.319
	0.005	SGDM	0.3008	10.00	0.666	4.722	0.361
		ADAM	0.2338	10.17	1.530	9.444	0.847
24	0.00001	SGDM	1.3658	38.26	30.969	36.472	32.122
		ADAM	0.4772	57.42	45.392	49.330	47.431
	0.00005	SGDM	1.1611	54.64	51.796	57.306	55.889
		ADAM	0.7211	59.47	52.799	59.652	56.463
	0.0001	SGDM	1.3350	58.77	52.867	56.043	54.927
		ADAM	0.1209	9.68	1.638	1.638	0.903
	0.0005	SGDM	2.2888	60.47	57.789	61.694	59.648
		ADAM	0.2597	10.01	2.160	11.944	1.198
	0.001	SGDM	19.1042	46.75	39.269	43.170	41.476
		ADAM	0.1559	10.01	1.634	9.444	0.903
	0.005	SGDM	0.1629	10.09	1.078	6.944	0.590
		ADAM	0.3708	10.24	1.837	8.615	1.031
28	0.00001	SGDM	2.8993	36.81	14.667	14.000	16.000
		ADAM	0.4116	57.33	32.667	32.000	34.000
	0.00005	SGDM	0.9147	54.74	34.667	33.167	37.667
		ADAM	20.0329	50.07	36.667	39.000	37.250

Table 1. *Cont.*

BS	LR	Optimizer	Standard Deviations (%)	Validation Accuracy (%)	F1-Score (%)	Recall (%)	Precision (%)
28	0.0001	SGDM	1.3348	56.45	38.167	39.167	40.333
		ADAM	0.1785	10.02	2.000	5.000	1.250
	0.0005	SGDM	2.1328	58.73	30.556	33.000	30.167
		ADAM	0.1679	9.96	0.000	0.000	0.000
	0.001	SGDM	19.4801	49.05	26.190	27.524	25.524
		ADAM	0.2318	9.80	0.000	0.000	0.000
	0.005	SGDM	0.1875	10.02	2.667	6.667	1.667
		ADAM	0.2691	9.94	0.000	0.000	0.000
32	0.00001	SGDM	1.8973	35.88	22.401	26.796	23.499
		ADAM	0.9375	56.38	42.033	46.230	42.611
	0.00005	SGDM	0.9352	52.98	48.248	51.367	55.033
		ADAM	2.5092	59.13	46.719	51.293	48.148
	0.0001	SGDM	1.2391	57.47	47.486	50.344	50.344
		ADAM	0.1832	10.00	1.868	9.444	1.059
	0.0005	SGDM	0.2294	59.84	49.280	53.456	51.630
		ADAM	0.1931	10.14	1.866	9.444	1.042
	0.001	SGDM	1.7214	58.68	43.959	49.241	44.963
		ADAM	0.2317	9.88	3.444	11.944	2.049
	0.005	SGDM	0.1667	9.98	1.634	9.444	0.903
		ADAM	0.1481	9.76	1.863	10.743	1.125

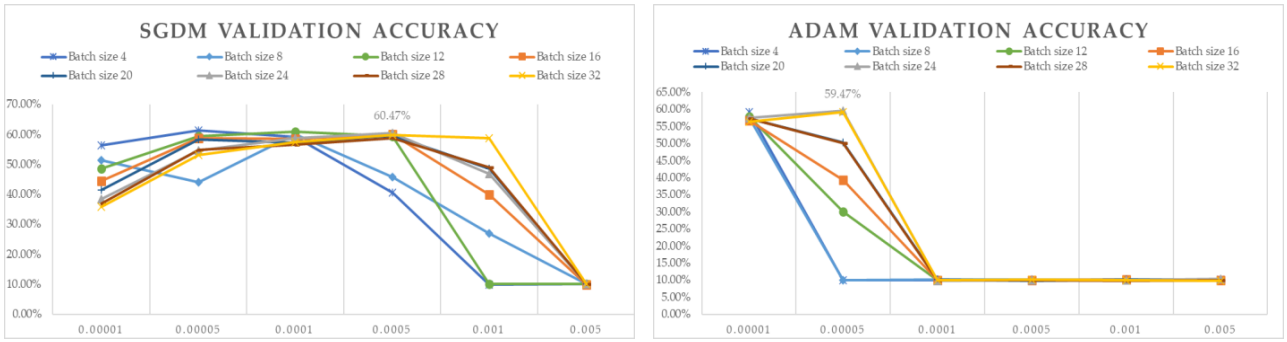
Table A2 in Appendix B details all the results of GridSearchCV and 5-fold cross-validation for various combinations of optimizers and hyperparameters, including mean validation accuracy, F1-score, precision, and recall.

In addition, the distribution of the results collected by the optimizer SGDM and ADAM is shown on the new hybrid model retrained on the CIFAR-10 dataset. On the left side of the table, the distribution of the measurement accuracy for a given BS ranges from 0.00001 to 0.005 for each specific LR. On the right side of the table, the distribution of the validation accuracy, F1-score, precision, and recall for a given LR range starting from 4 to 32 for each specific BS is shown.

When using SGDM with BS = 24 and LR = 0.0005, a maximum accuracy of 60.474%, an F1-score of 57.79%, a recall of 61.6944%, and a precision of 59.6481% were observed. In ADAM, the maximum accuracy = 59.47% was observed for BS = 24 and LR = 0.0005, while the maximum F1-score was 52.8%, recall was 59.6519%, and precision was 56.463%. Thus, using our new hybrid model on CIFAR-10, SGDM has better performance compared to ADAM as it achieves the maximum accuracy and F1-score, while also performing better in terms of recall and precision.

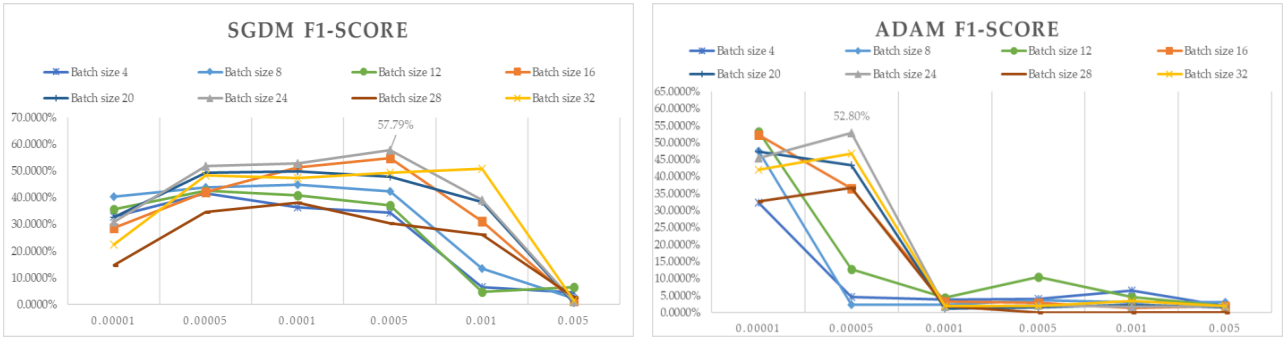
Figure 7a,b, Figure 8a,b, Figure 9a,b and Figure 10a,b depict the resulting curves of the validating accuracy, F1-score, recall, and precision for all parameters of SGDM and ADAM, respectively. The numerical labels of the best-performing dataset will be labeled with the specific values of BS = 24 and LR = 0.0005 in the SGDM optimizer and BS = 24 and LR = 0.00005 in the ADAM optimizer.

When using the SGDM optimizer, we observed that the difference in validation accuracy between different batch sizes was not significant when the learning rate was less than or equal to 0.005. However, when the learning rate was greater than or equal to 0.005, the difference in validation accuracy was more sensitive to changes in the learning rate. The F1-score, recall, and precision remained regular and stable across different batch size combinations.



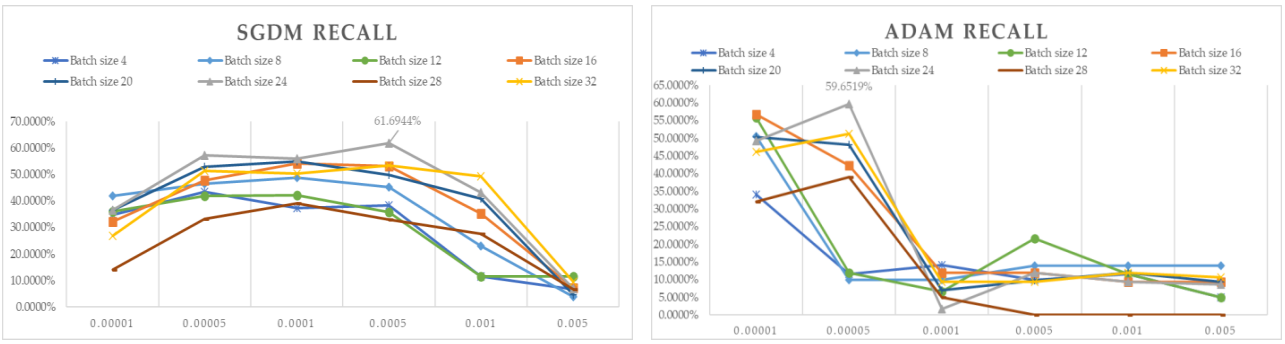
(**a**) (**b**)

Figure 7. Validation accuracy. (**a**) SGDM optimizer. (**b**) ADAM optimizer.



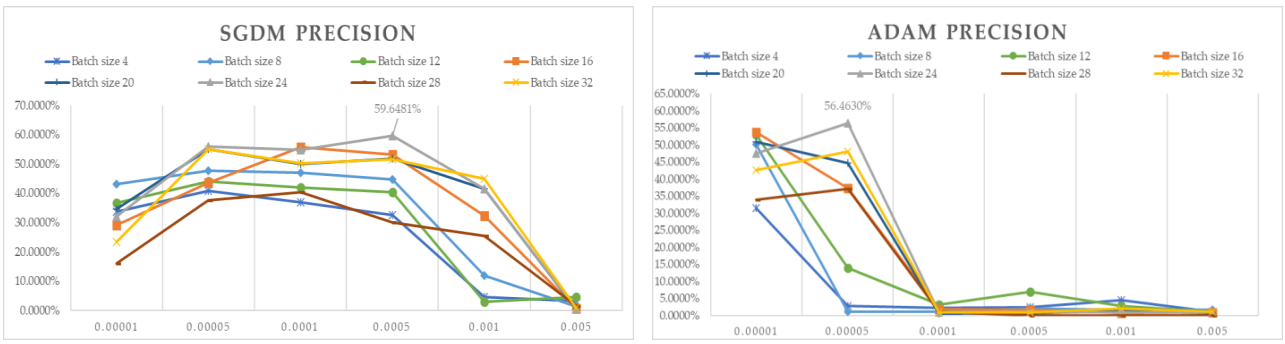
(**a**) (**b**)

Figure 8. F1-score. (**a**) SGDM optimizer. (**b**) ADAM optimizer.



(**a**) (**b**)

Figure 9. Recall. (**a**) SGDM optimizer. (**b**) ADAM optimizer.



(**a**) (**b**)

Figure 10. Precision. (**a**) SGDM optimizer. (**b**) ADAM optimizer.

When using the ADAM optimizer, we found that the difference in validation accuracy between different batch sizes was most significant when the learning rate was set to 0.0005. However, when the learning rate was greater than 0.001, the change in validation accuracy was negligible. The F1-score, recall, and precision showed some changes but not significant ones. Previous research has shown the use of an exponential moving average of the squares of the gradients generated by previous iterations [44]. This moving average is used to scale the current gradient after taking the square root of the average to update the weights. The contribution of the exponential mean is positive, and this approach should prevent the learning rate from becoming nearly infinitesimal during the learning process, which is a key drawback of the ADAM optimizer. However, the short-term memory capacity of this gradient becomes an obstacle in other cases. During the convergence of the ADAM optimizer to a suboptimal solution, it has been observed that some small batches of data provide large and informative gradients. Since these small batches occur very rarely, exponential averaging will reduce their impact. As a result, the ADAM optimizer corrects the gradient only when the learning rate is high, which can cause the algorithm to converge to suboptimal minima or even fail to converge, resulting in skipping local minima. The derivative can become too large, resulting in an infinite loss. This shows that ADAM does not generalize as well as SGDM.

3.2. Ablation Study between the Novel Hybrid Model and Original BEiT Model

We trained and tested the original BEiT model for 50 epochs on the CIFAR-10 dataset using the official default hyperparameters and optimizer configuration (batch size = 64, optimizer = ADAM, optimizer Epsilon = 1×10^{-8}, and learning rate = 5×10^{-4}). We then trained and tested our hybrid model for 50 epochs on the same dataset using the optimal configuration (batch size = 24, optimizer = SGDM, and learning rate = 5×10^{-4}). Table 2 shows the loss and test accuracy for each epoch and the test accuracy for both models. Figure 11 illustrates the process of training CIFAR-10 in the original BEiT model.

Table 2. The loss, testing accuracy for each epoch, and test accuracy for the original BEiT model and novel hybrid model.

Epoch	Original BEiT Model			Novel Hybrid Model			Epoch	Original BEiT Model			Novel Hybrid Model		
	Loss	Training Accuracy	Testing Accuracy	Loss	Training Accuracy	Testing Accuracy		Loss	Training Accuracy	Testing Accuracy	Loss	Training Accuracy	Testing Accuracy
0	4.457	4.00%	4.27%	1.987	24.23%	37.80%	25	3.078	32.80%	44.57%	0.000	100.00%	64.70%
1	4.224	7.47%	8.93%	1.547	43.26%	47.09%	26	3.059	33.41%	44.72%	0.000	100.00%	64.72%
2	4.121	9.49%	11.64%	1.353	51.38%	54.00%	27	3.041	34.02%	45.54%	0.000	100.00%	64.70%
3	4.066	10.49%	15.29%	1.211	56.45%	56.46%	28	3.045	34.07%	45.94%	0.000	100.00%	64.70%
4	4.026	11.27%	18.78%	1.085	61.25%	59.58%	29	3.034	34.12%	47.23%	0.000	100.00%	64.66%
5	3.973	12.52%	20.79%	0.956	66.00%	61.15%	30	2.826	34.67%	47.23%	0.000	100.00%	64.63%
6	3.913	13.60%	24.13%	0.837	70.27%	62.09%	31	2.780	35.00%	48.02%	0.000	100.00%	64.60%
7	3.856	14.74%	25.46%	0.715	74.53%	63.20%	32	2.735	35.33%	48.64%	0.000	100.00%	64.57%
8	3.803	15.94%	27.54%	0.588	79.06%	63.51%	33	2.689	35.66%	48.89%	0.000	100.00%	64.54%
9	3.747	17.04%	29.12%	0.468	83.28%	61.87%	34	2.643	35.99%	48.89%	0.000	100.00%	64.51%
10	3.710	17.90%	30.36%	0.355	87.26%	61.76%	35	2.597	36.32%	49.18%	0.000	100.00%	64.48%
11	3.651	18.94%	32.30%	0.258	90.74%	61.89%	36	2.551	36.65%	50.02%	0.000	100.00%	64.45%
12	3.608	20.04%	33.48%	0.191	93.38%	61.89%	37	2.505	36.98%	50.02%	0.000	100.00%	64.42%
13	3.561	21.15%	34.45%	0.150	94.72%	62.38%	38	2.459	37.31%	50.02%	0.000	100.00%	64.39%
14	3.517	22.23%	35.51%	0.105	62.38%	62.92%	39	2.413	37.64%	50.39%	0.000	100.00%	64.36%
15	3.476	23.36%	36.29%	0.084	97.10%	62.20%	40	2.368	37.97%	50.63%	0.000	100.00%	64.37%
16	3.423	24.35%	37.22%	0.072	97.48%	61.98%	41	2.322	38.30%	50.98%	0.000	100.00%	64.40%
17	3.379	25.34%	38.95%	0.055	98.17%	62.96%	42	2.276	38.63%	50.98%	0.000	100.00%	64.41%
18	3.332	26.46%	39.29%	0.045	98.53%	62.57%	43	2.230	38.96%	51.41%	0.000	100.00%	64.38%
19	3.286	27.87%	40.51%	0.041	98.67%	62.69%	44	2.184	39.29%	51.41%	0.000	100.00%	64.38%
20	3.243	28.71%	41.20%	0.019	99.41%	63.57%	45	2.138	39.62%	51.63%	0.000	100.00%	64.39%
21	3.193	29.72%	41.71%	0.011	99.68%	63.47%	46	2.092	39.95%	51.63%	0.000	100.00%	64.42%
22	3.147	30.53%	42.28%	0.005	99.86%	64.37%	47	2.046	40.28%	51.65%	0.000	100.00%	64.43%
23	3.100	31.30%	43.16%	0.001	100.00%	64.78%	48	1.977	40.61%	51.65%	0.000	100.00%	64.42%
24	3.054	32.42%	43.91%	0.000	100.00%	64.67%	49	1.928	40.94%	51.65%	0.000	100.00%	64.42%

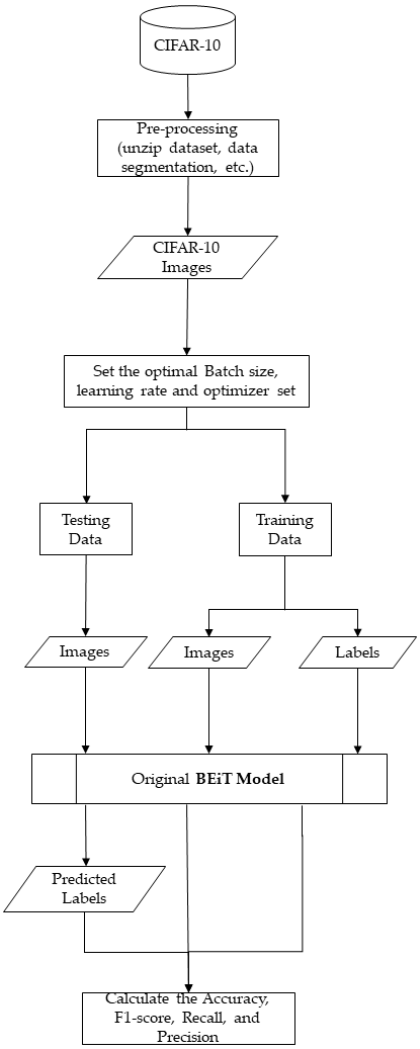


Figure 11. Process of training CIFAR-10 in the original BEiT model.

During training, the new hybrid model achieved 100% accuracy in 23 calendar hours, with loss dropping to 0. During validation, overfitting occurred in 10 epochs, with little improvement in accuracy during the subsequent validation process. On the other hand, the original BEiT model consistently improved in accuracy and decreased in loss during the training period. During validation, the original model never overfitted, but the performance improvement became smaller and smaller as the epochs increased. Due to the nature of cross-modality transfer learning, our model is pre-trained in the source domain using a completely different dataset from the target domain, which is a necessary condition for cross-modality transfer learning. In the comparison session, we do not compare the training accuracy of the two models but rather the testing accuracy. From the training results, the accuracy of our new hybrid model at the beginning of training was 12.77% higher compared to the original BEiT model. This is mainly due to pre-training; as the number of training sessions increased, both the original BEiT model and our hybrid model showed overfitting, but our hybrid model showed overfitting earlier, which made the difference between the accuracy of the original BEiT model and our model smaller. We performed Wilcoxon rank-sum tests between the novel hybrid model and the original BEiT model using training accuracy and testing accuracy. The null hypothesis H_0: accuracy of the novel hybrid model < accuracy of the original BEiT model is being rejected for all experimental settings (Table 2). Therefore, it is concluded that the novel hybrid model is statistically outperforming the original BEiT model. Figure 12 compares the accuracy of the two models tested over 50 epochs. The graph clearly shows that our model appears

to overfit earlier and that the difference between the accuracy of the two models becomes smaller and smaller until they both seem to overfit.

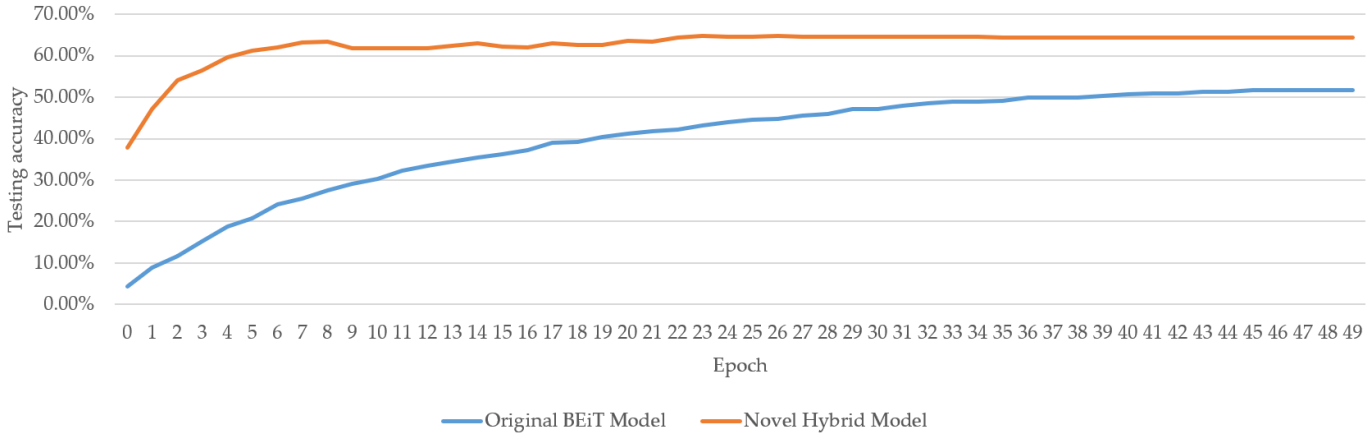


Figure 12. Comparison between the original BEiT model and the novel hybrid model.

4. Conclusions and Future Works

In this work, we propose a cross-modal transfer learning algorithm from the text domain to the image domain for image classification problems to solve tasks in the image classification domain. In the first phase of our work, two pre-trained models from different domains are trained on different source domains, and a new hybrid model is designed based on them. In the second phase of the work, we used GridSearchCV and 5-fold cross-validation to determine the best combination of hyperparameters by evaluating the validation accuracy, F1-score, precision, and recall of the model for different combinations. The results of the experiments not only allowed us to select the most efficient hyperparameters from various combinations of optimizers (SGDM, ADAM) and hyperparameters but also showed us that the optimizers and the two hyperparameters (BS and LR) had a significant impact on our model. In addition to these results, after several comparisons of BS and LR, we found that each hyperparameter affected our model's performance independently, suggesting that trade-offs should be made in the selection of BS and LR to obtain the highest F1-score. In the third stage, after our tests, we showed that, compared to the traditional BEiT model, the new hybrid model we designed had a higher accuracy.

It is worth noting that CMTL can facilitate knowledge transfer between the source and target domains of different modalities (low similarity between domains), where some knowledge cannot be learned from traditional transfer learning (domains with high similarity) [16,45,46]. Therefore, a comparison with non-CMTL approaches is not included in Section 3. Intuitively, combining traditional transfer learning with CMTL will further enhance the performance of the target model because more knowledge (from similar and dissimilar source domains) can be transferred, given that the issue of negative transfer is suppressed. We have thus suggested future work in this area. In future work, we would like to consider the application of migration learning to more different pre-trained models of text domains for image classification tasks, allowing a broader range of application scenarios for migration learning to occur. We believe that it is possible to study the effect of different layers on the results by adjusting the number of layers of the retained or frozen pre-trained model to study the importance of the last few layers in the overall model as well as the performance of the model on new datasets by reducing or increasing the number of layers in which the original model is retained, an approach that is considered an interesting direction for improving the effectiveness of migration learning in the future. Indeed, in addition to the text domain, many different source domains can be migrated to the image domain. In the future, higher accuracy can be achieved in the image classification domain by migrating to other domains. Furthermore, in our work, the evaluation of batch sizes larger than 32 is a current limitation due to GPU performance limitations. More analysis can

be conducted to evaluate the performance of the novel hybrid model using other datasets, such as the Visual Question Answering (VQA) 2.0 dataset [47].

Author Contributions: Formal analysis, J.L., K.T.C. and L.-K.L.; investigation, J.L., K.T.C. and L.-K.L.; methodology, J.L.; validation, J.L., K.T.C. and L.-K.L.; visualization, J.L.; writing—original draft, J.L., K.T.C. and L.-K.L.; writing—review and editing, J.L., K.T.C. and L.-K.L. All authors have read and agreed to the published version of the manuscript.

Funding: The work described in this paper was supported by the Katie Shu Sui Pui Charitable Trust—Research Training Fellowship (KSRTF/2022/07).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1. The table explains the detailed structure of our new hybrid model, including the types of layers, output shapes, and parameters, in order from left column to right column, and concludes with a summary of the model's parameters and sizes.

Layer (Type)	Output Shape	No. of Params	Layer (Type)	Output Shape	No. of Params
Conv2d-1	[−1, 64, 32, 32]	1792	Linear-127	[−1, 512, 768]	590,592
Conv2d-3	[−1, 128, 32, 32]	73,856	Linear-128	[−1, 512, 768]	590,592
Conv2d-5	[−1, 384, 32, 32]	442,752	Linear-129	[−1, 512, 768]	590,592
Linear-7	[−1, 512, 768]	590,592	Linear-130	[−1, 512, 768]	590,592
Linear-8	[−1, 512, 768]	590,592	Linear-131	[−1, 512, 768]	590,592
Linear-9	[−1, 512, 768]	590,592	Linear-132	[−1, 512, 768]	590,592
Linear-10	[−1, 512, 768]	590,592	Linear-135	[−1, 512, 768]	590,592
Linear-11	[−1, 512, 768]	590,592	Linear-136	[−1, 512, 768]	590,592
Linear-12	[−1, 512, 768]	590,592	LayerNorm-139	[−1, 512, 768]	1536
Linear-15	[−1, 512, 768]	590,592	LayerNorm-140	[−1, 512, 768]	1536
Linear-16	[−1, 512, 768]	590,592	Linear-141	[−1, 512, 3072]	2,362,368
LayerNorm-19	[−1, 512, 768]	1536	Linear-142	[−1, 512, 3072]	2,362,368
LayerNorm-20	[−1, 512, 768]	1536	Linear-145	[−1, 512, 768]	2,360,064
Linear-21	[−1, 512, 3072]	2,362,368	Linear-146	[−1, 512, 768]	2,360,064
Linear-22	[−1, 512, 3072]	2,362,368	LayerNorm-149	[−1, 512, 768]	1536
Linear-25	[−1, 512, 768]	2,360,064	LayerNorm-150	[−1, 512, 768]	1536
Linear-26	[−1, 512, 768]	2,360,064	Linear-151	[−1, 512, 768]	590,592
LayerNorm-29	[−1, 512, 768]	1536	Linear-152	[−1, 512, 768]	590,592
LayerNorm-30	[−1, 512, 768]	1536	Linear-153	[−1, 512, 768]	590,592
Linear-31	[−1, 512, 768]	590,592	Linear-154	[−1, 512, 768]	590,592
Linear-32	[−1, 512, 768]	590,592	Linear-155	[−1, 512, 768]	590,592
Linear-33	[−1, 512, 768]	590,592	Linear-156	[−1, 512, 768]	590,592
Linear-34	[−1, 512, 768]	590,592	Linear-159	[−1, 512, 768]	590,592
Linear-35	[−1, 512, 768]	590,592	Linear-160	[−1, 512, 768]	590,592
Linear-36	[−1, 512, 768]	590,592	LayerNorm-163	[−1, 512, 768]	1536
Linear-39	[−1, 512, 768]	590,592	LayerNorm-164	[−1, 512, 768]	1536
Linear-40	[−1, 512, 768]	590,592	Linear-165	[−1, 512, 3072]	2,362,368
LayerNorm-43	[−1, 512, 768]	1536	Linear-166	[−1, 512, 3072]	2,362,368
LayerNorm-44	[−1, 512, 768]	1536	Linear-169	[−1, 512, 768]	2,360,064
Linear-45	[−1, 512, 3072]	2,362,368	Linear-170	[−1, 512, 768]	2,360,064
Linear-46	[−1, 512, 3072]	2,362,368	LayerNorm-173	[−1, 512, 768]	1536
Linear-49	[−1, 512, 768]	2,360,064	LayerNorm-174	[−1, 512, 768]	1536
Linear-50	[−1, 512, 768]	2,360,064	Linear-175	[−1, 512, 768]	590,592
LayerNorm-53	[−1, 512, 768]	1536	Linear-176	[−1, 512, 768]	590,592
LayerNorm-54	[−1, 512, 768]	1536	Linear-177	[−1, 512, 768]	590,592
Linear-55	[−1, 512, 768]	590,592	Linear-178	[−1, 512, 768]	590,592
Linear-56	[−1, 512, 768]	590,592	Linear-179	[−1, 512, 768]	590,592

Table A1. *Cont.*

Layer (Type)	Output Shape	No. of Params	Layer (Type)	Output Shape	No. of Params
Linear-57	[−1, 512, 768]	590,592	Linear-180	[−1, 512, 768]	590,592
Linear-58	[−1, 512, 768]	590,592	Linear-183	[−1, 512, 768]	590,592
Linear-59	[−1, 512, 768]	590,592	Linear-184	[−1, 512, 768]	590,592
Linear-60	[−1, 512, 768]	590,592	LayerNorm-187	[−1, 512, 768]	1536
Linear-63	[−1, 512, 768]	590,592	LayerNorm-188	[−1, 512, 768]	1536
Linear-64	[−1, 512, 768]	590,592	Linear-189	[−1, 512, 3072]	2,362,368
LayerNorm-67	[−1, 512, 768]	1536	Linear-190	[−1, 512, 3072]	2,362,368
LayerNorm-68	[−1, 512, 768]	1536	Linear-193	[−1, 512, 768]	2,360,064
Linear-69	[−1, 512, 3072]	2,362,368	Linear-194	[−1, 512, 768]	2,360,064
Linear-70	[−1, 512, 3072]	2,362,368	LayerNorm-197	[−1, 512, 768]	1536
Linear-73	[−1, 512, 768]	2,360,064	LayerNorm-198	[−1, 512, 768]	1536
Linear-74	[−1, 512, 768]	2,360,064	Linear-199	[−1, 512, 768]	590,592
LayerNorm-77	[−1, 512, 768]	1536	Linear-200	[−1, 512, 768]	590,592
LayerNorm-78	[−1, 512, 768]	1536	Linear-201	[−1, 512, 768]	590,592
Linear-79	[−1, 512, 768]	590,592	Linear-202	[−1, 512, 768]	590,592
Linear-80	[−1, 512, 768]	590,592	Linear-203	[−1, 512, 768]	590,592
Linear-81	[−1, 512, 768]	590,592	Linear-204	[−1, 512, 768]	590,592
Linear-82	[−1, 512, 768]	590,592	Linear-207	[−1, 512, 768]	590,592
Linear-83	[−1, 512, 768]	590,592	Linear-208	[−1, 512, 768]	590,592
Linear-84	[−1, 512, 768]	590,592	LayerNorm-211	[−1, 512, 768]	1536
Linear-87	[−1, 512, 768]	590,592	LayerNorm-212	[−1, 512, 768]	1536
Linear-88	[−1, 512, 768]	590,592	Linear-213	[−1, 512, 3072]	2,362,368
LayerNorm-91	[−1, 512, 768]	1536	Linear-214	[−1, 512, 3072]	2,362,368
LayerNorm-92	[−1, 512, 768]	1536	Linear-217	[−1, 512, 768]	2,360,064
Linear-93	[−1, 512, 3072]	2,362,368	Linear-218	[−1, 512, 768]	2,360,064
Linear-94	[−1, 512, 3072]	2,362,368	LayerNorm-221	[−1, 512, 768]	1536
Linear-97	[−1, 512, 768]	2,360,064	LayerNorm-222	[−1, 512, 768]	1536
Linear-98	[−1, 512, 768]	2,360,064	Linear-223	[−1, 512, 768]	590,592
LayerNorm-101	[−1, 512, 768]	1536	Linear-224	[−1, 512, 768]	590,592
LayerNorm-102	[−1, 512, 768]	1536	Linear-225	[−1, 512, 768]	590,592
Linear-103	[−1, 512, 768]	590,592	Linear-226	[−1, 512, 768]	590,592
Linear-104	[−1, 512, 768]	590,592	Linear-227	[−1, 512, 768]	590,592
Linear-105	[−1, 512, 768]	590,592	Linear-228	[−1, 512, 768]	590,592
Linear-106	[−1, 512, 768]	590,592	Linear-231	[−1, 512, 768]	590,592
Linear-107	[−1, 512, 768]	590,592	Linear-232	[−1, 512, 768]	590,592
Linear-108	[−1, 512, 768]	590,592	LayerNorm-235	[−1, 512, 768]	1536
Linear-111	[−1, 512, 768]	590,592	LayerNorm-236	[−1, 512, 768]	1536
Linear-112	[−1, 512, 768]	590,592	Linear-237	[−1, 512, 3072]	2,362,368
LayerNorm-115	[−1, 512, 768]	1536	Linear-238	[−1, 512, 3072]	2,362,368
LayerNorm-116	[−1, 512, 768]	1536	Linear-241	[−1, 512, 768]	2,360,064
Linear-117	[−1, 512, 3072]	2,362,368	Linear-242	[−1, 512, 768]	2,360,064
Linear-118	[−1, 512, 3072]	2,362,368	LayerNorm-245	[−1, 512, 768]	1536
Linear-121	[−1, 512, 768]	2,360,064	LayerNorm-246	[−1, 512, 768]	1536
Linear-122	[−1, 512, 768]	2,360,064	LayerNorm-247	[−1, 197, 768]	1536
LayerNorm-125	[−1, 512, 768]	1536	Linear-249	[−1, 197, 768]	590,592
LayerNorm-126	[−1, 512, 768]	1536	LayerNorm-252	[−1, 197, 768]	1536
Total params: 152,928,522			Linear−253	[−1, 197, 3072]	2,362,368
Trainable params: 152,928,522			Linear−256	[−1, 197, 768]	2,360,064
Non-trainable params: 0			LayerNorm−259	[−1, 197, 768]	1536
Input size (MB): 0.011719			Linear−261	[−1, 197, 768]	590,592
Forward/backward pass size (MB): 1562.278091			LayerNorm−264	[−1, 197, 768]	1536
Params size (MB): 583.376015			Linear−265	[−1, 197, 3072]	2,362,368
Estimated Total Size (MB): 2145.665825			Linear−268	[−1, 197, 768]	2,360,064
			Linear−271	[−1, 10]	20,490

Appendix B

Table A2. This table shows all results for GridSearchCV and 5-fold cross-validation for various combinations of optimizers and hyperparameters, including mean validation accuracy, F1-score, precision, and recall.

SGDM													
Batch Size	**LR**	**Fold**	**Val Accuracy**	**F1-Score**	**Recall**	**Precision**	**Batch Size**	**LR**	**Fold**	**Val Accuracy**	**F1-Score**	**Recall**	**Precision**
4	0.00001	1	53.52%	33.33%	40.00%	30.00%	8	0.00001	1	49.53%	43.33%	45.83%	43.75%
		2	57.86%	33.33%	30.00%	40.00%			2	51.91%	61.90%	64.29%	64.29%
		3	57.3%	66.67%	75.00%	62.50%			3	52.70%	63.89%	66.67%	66.67%
		4	55.7%	20.00%	20.00%	20.00%			4	51.87%	8.33%	6.25%	12.50%
		5	57.22%	11.11%	8.33%	16.67%			5	51.13%	24.76%	26.19%	28.46%
		Mean	56.32%	32.89%	34.67%	33.83%			Mean	51.43%	40.44%	41.85%	43.15%
	0.00005	1	62.00%	33.33%	33.33%	33.33%		0.00005	1	61.63%	54.76%	61.90%	57.14%
		2	60.55%	50.00%	50.00%	50.00%			2	61.62%	52.38%	57.14%	57.14%
		3	61.33%	66.67%	75.00%	62.50%			3	60.74%	74.44%	75.00%	77.78%
		4	60.01%	33.33%	33.33%	33.33%			4	60.70%	23.81%	28.57%	21.43%
		5	62.68%	25.00%	25.00%	25.00%			5	61.98%	14.58%	10.42%	25.00%
		Mean	61.31%	41.67%	43.33%	40.83%			Mean	44.00%	44.00%	46.61%	47.70%
	0.0001	1	56.5%	16.67%	16.67%	16.67%		0.0001	1	61.92%	34.26%	33.33%	35.71%
		2	59.8%	50.00%	50.00%	50.00%			2	61.26%	40.48%	42.86%	42.86%
		3	62.45%	41.67%	50.00%	37.50%			3	60.67%	55.56%	58.33%	58.33%
		4	58.63%	60.00%	60.00%	60.00%			4	53.09%	28.57%	35.71%	31.43%
		5	58.36%	13.33%	10.00%	20.00%			5	63.20%	66.00%	73.33%	66.67%
		Mean	59.15%	36.33%	37.33%	36.83%			Mean	60.03%	44.98%	48.71%	47.00%
	0.0005	1	61.17%	33.33%	33.33%	33.33%		0.0005	1	55.87%	39.58%	41.67%	43.75%
		2	62.00%	50.00%	50.00%	50.00%			2	56.74%	58.33%	56.25%	62.50%
		3	10.22%	0.00%	0.00%	0.00%			3	47.04%	80.00%	83.33%	77.78%
		4	59.19%	66.67%	75.00%	62.50%			4	59.18%	29.17%	25.00%	37.50%
		5	9.88%	22.22%	33.33%	16.67%			5	10.06%	4.44%	2.00%	2.50%
		Mean	40.49%	34.44%	38.33%	32.50%			Mean	45.78%	42.31%	45.25%	44.81%
	0.001	1	10.2%	0.00%	0.00%	0.00%		0.001	1	9.69%	0.00%	0.00%	0.00%
		2	9.92%	0.00%	0.00%	0.00%			2	45.19%	40.48%	50.00%	40.48%
		3	9.74%	10.00%	25.00%	6.25%			3	59.85%	19.05%	28.57%	14.29%
		4	10.01%	0.00%	0.00%	0.00%			4	9.98%	3.70%	16.67%	2.08%
		5	9.88%	22.22%	33.33%	16.67%			5	10.03%	4.44%	20.00%	2.50%
		Mean	9.95%	6.44%	11.67%	4.58%			Mean	26.95%	13.53%	23.05%	11.87%
	0.005	1	10.20%	0.00%	0.00%	0.00%		0.005	1	10.20%	0.00%	0.00%	0.00%
		2	10.33%	0.00%	0.00%	0.00%			2	10.33%	0.00%	0.00%	0.00%
		3	9.91%	0.00%	0.00%	0.00%			3	9.91%	0.00%	0.00%	0.00%
		4	9.68%	0.00%	0.00%	0.00%			4	9.68%	0.00%	0.00%	0.00%
		5	9.88%	22.22%	33.33%	16.67%			5	10.10%	10.91%	20.00%	7.50%
		Mean	10.00%	4.44%	6.67%	3.33%			Mean	10.04%	2.18%	4.00%	1.50%
12	0.00001	1	48.55%	33.33%	33.33%	33.33%	16	0.00001	1	44.38%	38.52%	43.33%	41.67%
		2	48.11%	66.67%	62.50%	75.00%			2	46.77%	46.87%	56.25%	45.00%
		3	47.46%	33.33%	40.00%	30.00%			3	44.78%	30.50%	32.50%	29.17%
		4	49.04%	20.00%	20.00%	20.00%			4	43.87%	14.00%	15.00%	15.83%
		5	49.87%	25.00%	25.00%	25.00%			5	42.51%	14.00%	14.33%	14.33%
		Mean	48.60%	35.67%	36.17%	36.67%			Mean	44.46%	28.78%	32.28%	29.20%
	0.00005	1	57.77%	60.00%	60.00%	60.00%		0.00005	1	58.00%	41.48%	48.89%	43.52%
		2	61.49%	100.00%	100.00%	100.00%			2	61.71%	66.93%	66.67%	69.44%
		3	59.22%	20.00%	20.00%	20.00%			3	58.89%	31.69%	44.44%	26.30%
		4	59.59%	20.00%	20.00%	20.00%			4	57.47%	27.67%	37.50%	29.17%
		5	58.08%	13.33%	10.00%	20.00%			5	57.58%	43.15%	41.85%	50.00%
		Mean	59.23%	42.67%	42.00%	44.00%			Mean	58.73%	42.18%	47.87%	43.69%
	0.0001	1	61.68%	14.29%	14.29%	14.29%		0.0001	1	57.37%	55.37%	62.22%	53.70%
		2	62.67%	66.67%	62.50%	75.00%			2	63.05%	59.26%	57.41%	62.96%
		3	58.44%	33.33%	40.00%	30.00%			3	61.01%	49.01%	56.48%	53.17%
		4	62.01%	65.21%	68.75%	65.83%			4	61.52%	45.00%	42.50%	50.00%
		5	59.87%	25.00%	25.00%	25.00%			5	59.38%	47.83%	52.59%	58.33%
		Mean	60.93%	40.90%	42.11%	42.02%			Mean	60.47%	51.29%	54.24%	55.63%
	0.0005	1	59.34%	33.33%	33.33%	33.33%		0.0005	1	63.73%	61.46%	70.00%	63.54%
		2	58.71%	33.33%	33.33%	33.33%			2	60.60%	41.67%	50.00%	37.50%
		3	59.36%	45.67%	42.50%	55.00%			3	57.44%	48.89%	52.78%	46.76%
		4	60.51%	40.00%	40.00%	40.00%			4	58.33%	45.67%	42.50%	55.00%
		5	58.44%	33.33%	30.00%	40.00%			5	60.22%	54.50%	50.74%	62.96%
		Mean	59.27%	37.13%	35.83%	40.33%			Mean	60.06%	50.44%	53.20%	53.15%

Table A2. *Cont.*

SGDM													
Batch Size	**LR**	**Fold**	**Val Accuracy**	**F1-Score**	**Recall**	**Precision**	**Batch Size**	**LR**	**Fold**	**Val Accuracy**	**F1-Score**	**Recall**	**Precision**
12	0.001	1	10.11%	10.00%	25.00%	6.25%	16	0.001	1	10.01%	5.95%	12.50%	3.91%
		2	10.14%	0.00%	0.00%	0.00%			2	58.75%	70.42%	70.83%	72.92%
		3	10.00%	0.00%	0.00%	0.00%			3	59.32%	34.05%	45.00%	30.00%
		4	9.73%	0.00%	0.00%	0.00%			4	60.89%	45.33%	47.50%	55.00%
		5	10.06%	13.33%	33.33%	8.33%			5	10.24%	0.00%	0.00%	0.00%
		Mean	10.01%	4.67%	11.67%	2.92%			Mean	39.84%	31.15%	35.17%	32.36%
	0.005	1	10.20%	0.00%	0.00%	0.00%		0.005	1	10.20%	0.00%	0.00%	0.00%
		2	10.33%	0.00%	0.00%	0.00%			2	9.92%	2.78%	12.50%	1.56%
		3	9.91%	0.00%	0.00%	0.00%			3	9.91%	1.31%	11.11%	0.69%
		4	9.98%	10.00%	25.00%	6.25%			4	9.68%	0.00%	0.00%	0.00%
		5	9.88%	22.22%	33.33%	16.67%			5	9.88%	2.78%	12.50%	1.56%
		Mean	10.06%	6.44%	11.67%	4.58%			Mean	9.92%	1.37%	7.22%	0.76%
20	0.00001	1	42.66%	35.19%	41.85%	33.33%	24	0.00001	1	40.49%	32.08%	43.75%	31.67%
		2	42.15%	59.26%	65.63%	62.50%			2	38.41%	52.28%	51.85%	60.00%
		3	40.08%	31.33%	37.50%	29.17%			3	38.54%	22.52%	32.50%	18.33%
		4	41.68%	19.60%	19.76%	29.00%			4	37.56%	18.33%	25.00%	18.33%
		5	40.76%	17.46%	18.15%	19.75%			5	36.32%	29.63%	29.26%	32.28%
		Mean	41.47%	32.57%	36.58%	34.75%			Mean	38.26%	30.97%	36.47%	32.12%
	0.00005	1	56.44%	58.81%	65.83%	58.33%		0.00005	1	55.67%	44.81%	57.78%	49.07%
		2	59.93%	67.10%	71.88%	76.88%			2	52.94%	63.81%	64.58%	71.67%
		3	58.99%	44.36%	47.50%	44.33%			3	54.86%	52.65%	58.33%	50.37%
		4	57.21%	31.67%	32.62%	38.33%			4	56.01%	28.00%	32.50%	33.33%
		5	58.30%	45.17%	46.17%	57.50%			5	53.71%	69.71%	73.33%	75.00%
		Mean	58.17%	49.42%	52.80%	55.08%			Mean	54.64%	51.80%	57.31%	55.89%
	0.0001	1	60.15%	56.16%	60.00%	60.37%		0.0001	1	59.66%	45.33%	51.00%	42.50%
		2	62.88%	63.65%	66.67%	62.04%			2	58.20%	67.56%	68.75%	73.75%
		3	59.50%	47.35%	51.85%	44.44%			3	60.64%	38.70%	47.22%	34.26%
		4	61.04%	37.12%	40.95%	39.17%			4	58.66%	52.00%	52.50%	56.67%
		5	62.16%	44.71%	55.74%	44.44%			5	56.70%	60.74%	60.74%	67.46%
		Mean	61.15%	49.80%	55.04%	50.09%			Mean	58.77%	52.87%	56.04%	54.93%
	0.0005	1	57.81%	53.41%	56.67%	55.56%		0.0005	1	59.01%	53.39%	60.00%	53.70%
		2	58.31%	54.29%	55.00%	59.17%			2	62.81%	65.21%	68.75%	65.83%
		3	63.36%	57.17%	60.00%	61.00%			3	62.92%	67.65%	72.22%	64.44%
		4	59.04%	37.00%	35.95%	41.67%			4	56.93%	37.00%	37.50%	46.67%
		5	58.73%	37.38%	41.83%	41.50%			5	60.70%	65.70%	70.00%	67.59%
		Mean	59.45%	47.85%	49.89%	51.78%			Mean	60.47%	57.79%	61.69%	59.65%
	0.001	1	57.07%	37.08%	42.50%	35.42%		0.001	1	59.05%	53.70%	56.67%	58.33%
		2	58.38%	61.57%	60.00%	69.17%			2	61.08%	57.78%	57.41%	62.96%
		3	9.86%	2.02%	11.11%	1.11%			3	9.86%	1.31%	11.11%	0.69%
		4	59.46%	40.33%	40.95%	46.67%			4	46.66%	38.89%	41.67%	38.89%
		5	57.43%	50.56%	50.19%	55.56%			5	57.11%	44.67%	49.00%	46.50%
		Mean	48.44%	38.31%	40.95%	41.58%			Mean	46.75%	39.27%	43.17%	41.48%
	0.005	1	9.73%	2.27%	12.50%	1.25%		0.005	1	10.20%	0.00%	0.00%	0.00%
		2	10.46%	0.00%	0.00%	0.00%			2	9.93%	2.78%	12.50%	1.56%
		3	9.91%	1.06%	11.11%	0.56%			3	9.86%	1.31%	11.11%	0.69%
		4	9.68%	0.00%	0.00%	0.00%			4	10.23%	1.31%	11.11%	0.69%
		5	10.24%	0.00%	0.00%	0.00%			5	10.24%	0.00%	0.00%	0.00%
		Mean	10.00%	0.67%	4.72%	0.36%			Mean	10.09%	1.08%	6.94%	0.59%
28	0.00001	1	31.34%	0.00%	0.00%	0.00%	32	0.00001	1	36.77%	21.11%	32.22%	23.52%
		2	37.39%	0.00%	0.00%	0.00%			2	39.12%	31.10%	37.50%	31.25%
		3	38.16%	40.00%	40.00%	40.00%			3	34.83%	27.46%	33.33%	24.81%
		4	39.94%	20.00%	20.00%	20.00%			4	34.98%	15.67%	15.00%	18.33%
		5	37.20%	13.33%	10.00%	20.00%			5	33.69%	16.67%	15.93%	19.58%
		Mean	36.81%	14.67%	14.00%	16.00%			Mean	35.88%	22.40%	26.80%	23.50%
	0.00005	1	56.21%	33.33%	33.33%	33.33%		0.00005	1	52.80%	55.24%	60.00%	66.67%
		2	53.35%	66.67%	62.50%	75.00%			2	52.40%	62.14%	62.50%	71.67%
		3	54.51%	20.00%	20.00%	20.00%			3	53.83%	39.52%	47.50%	35.83%
		4	54.93%	40.00%	40.00%	40.00%			4	51.65%	43.33%	42.50%	55.00%
		5	54.70%	13.33%	10.00%	20.00%			5	54.21%	41.00%	44.33%	46.00%
		Mean	54.74%	34.67%	33.17%	37.67%			Mean	52.98%	48.25%	51.37%	55.03%
	0.0001	1	61.51%	33.33%	33.33%	33.33%		0.0001	1	56.69%	57.22%	62.22%	59.26%
		2	59.69%	66.67%	62.50%	75.00%			2	58.72%	72.92%	75.00%	75.00%
		3	60.92%	40.00%	40.00%	40.00%			3	57.91%	29.10%	41.67%	23.15%
		4	58.24%	37.50%	50.00%	33.33%			4	55.45%	52.00%	47.50%	66.67%
		5	61.90%	13.33%	10.00%	20.00%			5	58.58%	26.19%	25.33%	35.00%
		Mean	60.45%	38.17%	39.17%	40.33%			Mean	57.47%	47.49%	50.34%	50.34%

Table A2. *Cont.*

SGDM													
Batch Size	**LR**	**Fold**	**Val Accuracy**	**F1-Score**	**Recall**	**Precision**	**Batch Size**	**LR**	**Fold**	**Val Accuracy**	**F1-Score**	**Recall**	**Precision**
28	0.0005	1	56.86%	16.67%	16.67%	16.67%	32	0.0005	1	59.85%	58.20%	60.00%	63.89%
		2	61.44%	50.00%	50.00%	50.00%			2	60.23%	57.30%	61.11%	54.63%
		3	63.29%	41.67%	50.00%	37.50%			3	59.80%	44.71%	50.00%	42.96%
		4	61.54%	33.33%	40.00%	30.00%			4	59.51%	46.67%	52.50%	48.33%
		5	60.50%	11.11%	8.33%	16.67%			5	59.82%	39.52%	43.67%	48.33%
		Mean	60.73%	30.56%	33.00%	30.17%			Mean	59.84%	49.28%	53.46%	51.63%
	0.001	1	58.72%	14.29%	14.29%	14.29%		0.001	1	60.31%	36.67%	42.22%	34.81%
		2	59.69%	50.00%	50.00%	50.00%			2	60.94%	37.50%	50.00%	33.33%
		3	58.04%	33.33%	40.00%	30.00%			3	58.44%	39.00%	43.33%	40.00%
		4	58.68%	33.33%	33.33%	33.33%			4	56.43%	40.33%	42.50%	46.67%
		5	10.10%	0.00%	0.00%	0.00%			5	57.29%	66.30%	68.15%	70.00%
		Mean	49.05%	26.19%	27.52%	25.52%			Mean	58.68%	43.96%	49.24%	44.96%
	0.005	1	10.20%	0.00%	0.00%	0.00%		0.005	1	10.20%	0.00%	0.00%	0.00%
		2	9.70%	13.33%	33.33%	8.33%			2	9.70%	2.78%	12.50%	1.56%
		3	9.91%	0.00%	0.00%	0.00%			3	9.91%	1.31%	11.11%	0.69%
		4	10.17%	0.00%	0.00%	0.00%			4	10.01%	1.31%	11.11%	0.69%
		5	10.10%	0.00%	0.00%	0.00%			5	10.06%	2.78%	12.50%	1.56%
		Mean	10.02%	2.67%	6.67%	1.67%			Mean	9.98%	1.63%	9.44%	0.90%
ADAM													
Batch size	**LR**	**Fold**	**Val Accuracy**	**F1-score**	**Recall**	**Precision**	**Batch size**	**LR**	**Fold**	**Val Accuracy**	**F1-score**	**Recall**	**Precision**
4	0.00001	1	58.63%	40.00%	40.00%	40.00%	8	0.00001	1	56.96%	63.81%	66.67%	64.29%
		2	65.19%	20.00%	20.00%	20.00%			2	57.84%	45.83%	43.75%	50.00%
		3	57.11%	41.67%	50.00%	37.50%			3	57.31%	40.00%	42.86%	38.10%
		4	57.33%	40.00%	40.00%	40.00%			4	57.54%	29.52%	35.71%	32.14%
		5	57.64%	20.00%	20.00%	20.00%			5	57.80%	58.33%	63.89%	66.67%
		Mean	59.18%	32.33%	34.00%	31.50%			Mean	57.49%	47.50%	50.58%	50.24%
	0.00005	1	9.73%	0.00%	0.00%	0.00%		0.00005	1	10.20%	0.00%	0.00%	0.00%
		2	10.17%	13.33%	33.33%	8.33%			2	10.14%	3.70%	16.67%	2.08%
		3	10.25%	10.00%	25.00%	6.25%			3	10.25%	3.70%	16.67%	2.08%
		4	9.68%	0.00%	0.00%	0.00%			4	9.73%	3.70%	16.67%	2.08%
		5	10.03%	0.00%	0.00%	0.00%			5	9.46%	0.00%	0.00%	0.00%
		Mean	9.97%	4.67%	11.67%	2.92%			Mean	9.96%	2.22%	10.00%	1.25%
	0.0001	1	9.94%	10.00%	25.00%	6.25%		0.0001	1	10.20%	0.00%	0.00%	0.00%
		2	10.17%	3.70%	16.67%	2.08%			2	9.93%	3.70%	16.67%	2.08%
		3	9.73%	3.70%	16.67%	2.08%			3	10.25%	3.70%	16.67%	2.08%
		4	9.46%	1.47%	12.50%	0.78%			4	10.40%	3.70%	16.67%	2.08%
		5	10.24%	0.00%	0.00%	0.00%			5	9.46%	0.00%	0.00%	0.00%
		Mean	9.91%	3.77%	14.17%	2.24%			Mean	10.05%	2.22%	10.00%	1.25%
	0.0005	1	9.94%	10.00%	25.00%	6.25%		0.0005	1	10.37%	3.70%	16.67%	2.08%
		2	9.92%	0.00%	0.00%	0.00%			2	9.70%	6.67%	16.67%	4.17%
		3	10.03%	0.00%	0.00%	0.00%			3	10.22%	0.00%	0.00%	0.00%
		4	10.40%	10.00%	25.00%	6.25%			4	10.40%	3.70%	16.67%	2.08%
		5	9.46%	0.00%	0.00%	0.00%			5	10.06%	4.44%	20.00%	2.50%
		Mean	9.95%	4.00%	10.00%	2.50%			Mean	10.15%	3.70%	14.00%	2.17%
	0.001	1	10.37%	10.00%	25.00%	6.25%		0.001	1	9.72%	3.70%	16.67%	2.08%
		2	9.60%	22.22%	33.33%	16.67%			2	10.46%	0.00%	0.00%	0.00%
		3	9.86%	0.00%	0.00%	0.00%			3	9.86%	3.70%	16.67%	2.08%
		4	10.44%	0.00%	0.00%	0.00%			4	9.73%	3.70%	16.67%	2.08%
		5	9.46%	0.00%	0.00%	0.00%			5	10.03%	4.44%	20.00%	2.50%
		Mean	9.95%	6.44%	11.67%	4.58%			Mean	9.96%	3.11%	14.00%	1.75%
	0.005	1	10.03%	0.00%	0.00%	0.00%		0.005	1	10.11%	3.70%	16.67%	2.08%
		2	10.14%	0.00%	0.00%	0.00%			2	10.33%	0.00%	0.00%	0.00%
		3	10.22%	0.00%	0.00%	0.00%			3	9.86%	3.70%	16.67%	2.08%
		4	10.40%	10.00%	25.00%	6.25%			4	9.73%	3.70%	16.67%	2.08%
		5	9.79%	0.00%	0.00%	0.00%			5	10.10%	4.44%	20.00%	2.50%
		Mean	10.12%	2.00%	5.00%	1.25%			Mean	10.03%	3.11%	14.00%	1.75%
12	0.00001	1	57.81%	33.33%	33.33%	33.33%	16	0.00001	1	56.61%	58.33%	62.22%	64.81%
		2	57.96%	100.00%	100.00%	100.00%			2	56.60%	66.55%	68.75%	67.71%
		3	59.22%	33.33%	40.00%	30.00%			3	56.23%	50.51%	50.44%	50.67%
		4	56.18%	60.00%	60.00%	60.00%			4	57.70%	41.00%	45.00%	38.33%
		5	57.68%	39.83%	46.00%	41.67%			5	56.55%	45.19%	57.78%	47.22%
		Mean	57.77%	53.30%	55.87%	53.00%			Mean	56.74%	52.31%	56.84%	53.75%
	0.00005	1	9.73%	0.00%	0.00%	0.00%		0.00005	1	60.28%	41.00%	46.00%	41.00%
		2	9.82%	0.00%	0.00%	0.00%			2	59.55%	70.71%	75.00%	73.96%
		3	10.00%	0.00%	0.00%	0.00%			3	57.40%	65.33%	66.67%	68.33%
		4	58.26%	50.00%	50.00%	50.00%			4	9.44%	4.44%	11.11%	2.78%
		5	61.87%	13.33%	10.00%	20.00%			5	9.46%	1.47%	12.50%	0.78%
		Mean	29.94%	12.67%	12.00%	14.00%			Mean	39.23%	36.59%	42.26%	37.37%

Table A2. *Cont.*

SGDM												
Batch Size	**LR**	**Fold**	**Val Accuracy**	**F1-Score**	**Recall**	**Precision**	**Batch Size**	**LR**	**Fold**	**Val Accuracy**	**F1-Score**	**Recall**	**Precision**
12	0.0001	1	10.20%	0.00%	0.00%	0.00%	16	0.0001	1	9.94%	2.78%	12.50%	1.56%
		2	9.60%	22.22%	33.33%	16.67%			2	9.82%	2.78%	12.50%	1.56%
		3	10.03%	0.00%	0.00%	0.00%			3	10.25%	3.51%	11.11%	2.08%
		4	9.73%	0.00%	0.00%	0.00%			4	9.44%	4.44%	11.11%	2.78%
		5	10.41%	0.00%	0.00%	0.00%			5	9.88%	2.78%	12.50%	1.56%
		Mean	9.99%	4.44%	6.67%	3.33%			Mean	9.87%	3.26%	11.94%	1.91%
	0.0005	1	10.11%	10.00%	25.00%	6.25%		0.0005	1	10.01%	5.95%	12.50%	3.91%
		2	9.93%	0.00%	0.00%	0.00%			2	9.82%	2.78%	12.50%	1.56%
		3	9.74%	10.00%	25.00%	6.25%			3	9.91%	1.31%	11.11%	0.69%
		4	9.98%	10.00%	25.00%	6.25%			4	9.73%	2.47%	11.11%	1.39%
		5	9.88%	22.22%	33.33%	16.67%			5	9.93%	1.47%	12.50%	0.78%
		Mean	9.93%	10.44%	21.67%	7.08%			Mean	9.88%	2.80%	11.94%	1.67%
	0.001	1	10.20%	0.00%	0.00%	0.00%		0.001	1	10.20%	1.47%	12.50%	0.78%
		2	9.93%	0.00%	0.00%	0.00%			2	9.92%	2.78%	12.50%	1.56%
		3	9.86%	0.00%	0.00%	0.00%			3	9.86%	1.31%	11.11%	0.69%
		4	10.40%	10.00%	25.00%	6.25%			4	10.17%	1.31%	11.11%	0.69%
		5	10.06%	13.33%	33.33%	8.33%			5	10.41%	0.00%	0.00%	0.00%
		Mean	10.09%	4.67%	11.67%	2.92%			Mean	10.11%	1.37%	9.44%	0.75%
	0.005	1	10.37%	10.00%	25.00%	6.25%		0.005	1	9.94%	2.78%	12.50%	1.56%
		2	10.14%	0.00%	0.00%	0.00%			2	9.92%	2.78%	12.50%	1.56%
		3	9.91%	0.00%	0.00%	0.00%			3	9.96%	1.31%	11.11%	0.69%
		4	9.73%	0.00%	0.00%	0.00%			4	9.82%	0.00%	0.00%	0.00%
		5	9.93%	0.00%	0.00%	0.00%			5	9.95%	1.47%	12.50%	0.78%
		Mean	10.02%	2.00%	5.00%	1.25%			Mean	9.87%	1.99%	9.41%	0.91%
20	0.00001	1	57.32%	68.81%	69.58%	80.21%	24	0.00001	1	57.68%	39.83%	46.00%	41.67%
		2	59.61%	62.78%	62.96%	63.89%			2	57.35%	45.93%	50.00%	46.30%
		3	56.62%	61.48%	69.44%	59.26%			3	57.34%	43.46%	48.15%	41.48%
		4	56.34%	41.94%	37.38%	50.00%			4	56.63%	28.00%	30.00%	31.67%
		5	57.02%	2.27%	12.50%	1.25%			5	58.08%	69.74%	72.50%	76.04%
		Mean	57.38%	47.46%	50.37%	50.92%			Mean	57.42%	45.39%	49.33%	47.43%
	0.00005	1	60.15%	48.24%	49.00%	60.33%		0.00005	1	60.55%	63.54%	70.00%	66.67%
		2	61.36%	70.63%	71.13%	70.33%			2	60.09%	49.63%	51.85%	48.15%
		3	59.93%	42.95%	53.33%	38.17%			3	59.16%	68.52%	76.85%	72.22%
		4	60.65%	52.59%	56.61%	53.70%			4	58.80%	39.26%	47.22%	44.44%
		5	9.79%	2.47%	11.11%	1.39%			5	58.76%	43.05%	52.33%	50.83%
		Mean	50.38%	43.38%	48.24%	44.79%			Mean	59.47%	52.80%	59.65%	56.46%
	0.0001	1	9.72%	2.27%	12.50%	1.25%		0.0001	1	9.72%	1.47%	12.50%	0.78%
		2	10.33%	0.00%	0.00%	0.00%			2	9.82%	2.78%	12.50%	1.56%
		3	9.86%	2.02%	11.11%	1.11%			3	9.74%	2.47%	11.11%	1.39%
		4	10.23%	1.06%	11.11%	0.56%			4	9.68%	0.00%	0.00%	0.00%
		5	10.24%	0.00%	0.00%	0.00%			5	9.46%	1.47%	12.50%	0.78%
		Mean	10.08%	1.07%	6.94%	0.58%			Mean	9.68%	1.64%	1.64%	0.90%
	0.0005	1	9.72%	2.27%	12.50%	1.25%		0.0005	1	10.37%	2.78%	12.50%	1.56%
		2	9.70%	2.27%	12.50%	1.25%			2	9.92%	2.78%	12.50%	1.56%
		3	9.91%	1.06%	11.11%	0.56%			3	10.22%	1.31%	11.11%	0.69%
		4	9.68%	0.00%	0.00%	0.00%			4	9.73%	2.47%	11.11%	1.39%
		5	9.79%	2.27%	12.50%	1.25%			5	9.73%	1.47%	12.50%	0.78%
		Mean	9.76%	1.58%	9.72%	0.86%			Mean	10.01%	2.16%	11.94%	1.20%
	0.001	1	10.11%	3.26%	12.50%	1.88%		0.001	1	10.11%	2.78%	12.50%	1.56%
		2	10.14%	3.26%	12.50%	1.88%			2	9.82%	2.78%	12.50%	1.56%
		3	10.22%	2.90%	11.11%	1.67%			3	9.86%	1.31%	11.11%	0.69%
		4	10.17%	1.06%	11.11%	0.56%			4	10.01%	1.31%	11.11%	0.69%
		5	9.93%	1.19%	12.50%	0.63%			5	10.24%	0.00%	0.00%	0.00%
		Mean	10.11%	2.33%	11.94%	1.32%			Mean	10.01%	1.63%	9.44%	0.90%
	0.005	1	10.37%	2.27%	12.50%	1.25%		0.005	1	9.73%	2.78%	12.50%	1.56%
		2	10.46%	0.00%	0.00%	0.00%			2	10.17%	1.47%	12.50%	0.78%
		3	9.81%	1.06%	11.11%	0.56%			3	10.78%	4.44%	11.11%	2.78%
		4	10.17%	1.06%	11.11%	0.56%			4	9.94%	2.78%	12.50%	1.56%
		5	10.03%	3.26%	12.50%	1.88%			5	9.86%	1.31%	11.11%	0.69%
		Mean	10.17%	1.53%	9.44%	0.85%			Mean	10.24%	1.84%	8.61%	1.03%
28	0.00001	1	57.46%	40.00%	40.00%	40.00%	32	0.00001	1	54.92%	43.17%	46.00%	43.33%
		2	57.47%	50.00%	50.00%	50.00%			2	57.72%	39.31%	42.59%	45.19%
		3	56.73%	40.00%	40.00%	40.00%			3	56.87%	42.59%	55.56%	37.04%
		4	57.94%	20.00%	20.00%	20.00%			4	55.91%	42.00%	45.00%	40.00%
		5	57.05%	13.33%	10.00%	20.00%			5	56.46%	43.10%	42.00%	47.50%
		Mean	57.33%	32.67%	32.00%	34.00%			Mean	56.38%	42.03%	46.23%	42.61%
	0.00005	1	10.01%	10.00%	25.00%	6.25%		0.00005	1	60.60%	41.67%	51.11%	41.67%
		2	60.19%	50.00%	50.00%	50.00%			2	60.71%	42.67%	48.33%	44.17%
		3	60.61%	50.00%	50.00%	50.00%			3	65.97%	59.26%	68.52%	57.41%
		4	59.38%	40.00%	40.00%	40.00%			4	59.06%	30.00%	27.50%	35.00%
		5	60.15%	33.33%	30.00%	40.00%			5	59.31%	60.00%	61.00%	62.50%
		Mean	50.07%	36.67%	39.00%	37.25%			Mean	59.13%	46.72%	51.29%	48.15%

Table A2. *Cont.*

SGDM													
Batch Size	**LR**	**Fold**	**Val Accuracy**	**F1-Score**	**Recall**	**Precision**	**Batch Size**	**LR**	**Fold**	**Val Accuracy**	**F1-Score**	**Recall**	**Precision**
28	0.0001	1	10.03%	0.00%	0.00%	0.00%	32	0.0001	1	10.20%	0.00%	0.00%	0.00%
		2	9.92%	0.00%	0.00%	0.00%			2	9.70%	2.78%	12.50%	1.56%
		3	9.74%	10.00%	25.00%	6.25%			3	9.91%	1.31%	11.11%	0.69%
		4	10.17%	0.00%	0.00%	0.00%			4	10.17%	1.31%	11.11%	0.69%
		5	10.24%	0.00%	0.00%	0.00%			5	10.03%	3.95%	12.50%	2.34%
		Mean	10.02%	2.00%	5.00%	1.25%			Mean	10.00%	1.87%	9.44%	1.06%
	0.0005	1	10.20%	0.00%	0.00%	0.00%		0.0005	1	10.20%	0.00%	0.00%	0.00%
		2	9.93%	0.00%	0.00%	0.00%			2	9.82%	2.78%	12.50%	1.56%
		3	9.86%	0.00%	0.00%	0.00%			3	10.22%	1.31%	11.11%	0.69%
		4	9.73%	0.00%	0.00%	0.00%			4	10.40%	2.47%	11.11%	1.39%
		5	10.10%	0.00%	0.00%	0.00%			5	10.06%	2.78%	12.50%	1.56%
		Mean	9.96%	0.00%	0.00%	0.00%			Mean	10.14%	1.87%	9.44%	1.04%
	0.001	1	9.69%	0.00%	0.00%	0.00%		0.001	1	10.11%	2.78%	12.50%	1.56%
		2	9.82%	0.00%	0.00%	0.00%			2	9.93%	2.78%	12.50%	1.56%
		3	9.86%	0.00%	0.00%	0.00%			3	10.02%	4.44%	11.11%	2.78%
		4	10.17%	0.00%	0.00%	0.00%			4	9.44%	4.44%	11.11%	2.78%
		5	9.46%	0.00%	0.00%	0.00%			5	9.88%	2.78%	12.50%	1.56%
		Mean	9.80%	0.00%	0.00%	0.00%			Mean	9.88%	3.44%	11.94%	2.05%
	0.005	1	9.94%	0.00%	0.00%	0.00%		0.005	1	9.72%	1.47%	12.50%	0.78%
		2	10.17%	0.00%	0.00%	0.00%			2	10.17%	1.47%	12.50%	0.78%
		3	10.22%	0.00%	0.00%	0.00%			3	9.86%	1.31%	11.11%	0.69%
		4	9.92%	0.00%	0.00%	0.00%			4	9.94%	2.68%	12.42%	1.57%
		5	9.46%	0.00%	0.00%	0.00%			5	9.86%	2.14%	10.21%	1.38%
		Mean	9.94%	0.00%	0.00%	0.00%			Mean	9.76%	1.86%	10.74%	1.12%

References

1. Pouyanfar, S.; Sadiq, S.; Yan, Y.; Tian, H.; Tao, Y.; Reyes, M.P.; Shyu, M.-L.; Chen, S.-C.; Iyengar, S.S. A Survey on Deep Learning: Algorithms, Techniques, and Applications. *ACM Comput. Surv.* **2018**, *51*, 1–36. [CrossRef]
2. Zhu, W.; Braun, B.; Chiang, L.H.; Romagnoli, J.A. Investigation of Transfer Learning for Image Classification and Impact on Training Sample Size. *Chemom. Intell. Lab. Syst.* **2021**, *211*, 104269. [CrossRef]
3. Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.; Santamaría, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L. Review of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions. *J. Big Data* **2021**, *8*, 53. [PubMed]
4. Li, Q.; Cai, W.; Wang, X.; Zhou, Y.; Feng, D.D.; Chen, M. Medical Image Classification with Convolutional Neural Network. In Proceedings of the 2014 13th International Conference on Control Automation Robotics & Vision (ICARCV), Singapore, 10–12 December 2014; pp. 844–848.
5. Decherchi, S.; Pedrini, E.; Mordenti, M.; Cavalli, A.; Sangiorgi, L. Opportunities and Challenges for Machine Learning in Rare Diseases. *Front. Med.* **2021**, *8*, 1696. [CrossRef]
6. Han, J.; Zhang, Z.; Mascolo, C.; André, E.; Tao, J.; Zhao, Z.; Schuller, B.W. Deep Learning for Mobile Mental Health: Challenges and Recent Advances. *IEEE Signal Process. Mag.* **2021**, *38*, 96–105. [CrossRef]
7. Sovrano, F.; Palmirani, M.; Vitali, F. Combining Shallow and Deep Learning Approaches against Data Scarcity in Legal Domains. *Gov. Inf. Q.* **2022**, *39*, 101715. [CrossRef]
8. Morid, M.A.; Borjali, A.; Del Fiol, G. A Scoping Review of Transfer Learning Research on Medical Image Analysis Using ImageNet. *Comput. Biol. Med.* **2021**, *128*, 104115. [CrossRef]
9. Chui, K.T.; Gupta, B.B.; Jhaveri, R.H.; Chi, H.R.; Arya, V.; Almomani, A.; Nauman, A. Multiround transfer learning and modified generative adversarial network for lung cancer detection. *Int. J. Intell. Syst.* **2023**, *2023*, 6376275. [CrossRef]
10. Hussain, M.; Bird, J.J.; Faria, D.R. A Study on Cnn Transfer Learning for Image Classification. In *Advances in Computational Intelligence Systems: Contributions Proceedings of the 18th UK Workshop on Computational Intelligence, Nottingham, UK, 5–7 September 2018*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 191–202.
11. Salehi, A.W.; Khan, S.; Gupta, G.; Alabduallah, B.I.; Almjally, A.; Alsolai, H.; Siddiqui, T.; Mellit, A. A Study of CNN and Transfer Learning in Medical Imaging: Advantages, Challenges, Future Scope. *Sustainability* **2023**, *15*, 5930. [CrossRef]
12. Wang, Y.; Mori, G. Max-Margin Hidden Conditional Random Fields for Human Action Recognition. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 872–879.
13. Yao, A.; Gall, J.; Van Gool, L. A Hough Transform-Based Voting Framework for Action Recognition. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 2061–2068.
14. Xia, T.; Tao, D.; Mei, T.; Zhang, Y. Multiview Spectral Embedding. *IEEE Trans. Syst. Man Cybern. B* **2010**, *40*, 1438–1446.
15. Shao, L.; Zhu, F.; Li, X. Transfer Learning for Visual Categorization: A Survey. *IEEE Trans. Neural Netw. Learn. Syst.* **2014**, *26*, 1019–1034. [CrossRef] [PubMed]

16. Zhuang, F.; Qi, Z.; Duan, K.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; He, Q. A Comprehensive Survey on Transfer Learning. *Proc. IEEE* **2020**, *109*, 43–76.
17. Wang, Z.; Dai, Z.; Póczos, B.; Carbonell, J. Characterizing and Avoiding Negative Transfer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 11293–11302.
18. Chui, K.T.; Arya, V.; Band, S.S.; Alhalabi, M.; Liu, R.W.; Chi, H.R. Facilitating Innovation and Knowledge Transfer between Homogeneous and Heterogeneous Datasets: Generic Incremental Transfer Learning Approach and Multidisciplinary Studies. *J. Innov. Knowl.* **2023**, *8*, 100313. [CrossRef]
19. Niu, S.; Jiang, Y.; Chen, B.; Wang, J.; Liu, Y.; Song, H. Cross-Modality Transfer Learning for Image-Text Information Management. *ACM Trans. Manag. Inf. Syst.* **2021**, *13*, 1–14. [CrossRef]
20. Lei, H.; Han, T.; Zhou, F.; Yu, Z.; Qin, J.; Elazab, A.; Lei, B. A Deeply Supervised Residual Network for HEp-2 Cell Classification via Cross-Modal Transfer Learning. *Pattern Recognit.* **2018**, *79*, 290–302. [CrossRef]
21. Vununu, C.; Lee, S.-H.; Kwon, K.-R. A Classification Method for the Cellular Images Based on Active Learning and Cross-Modal Transfer Learning. *Sensors* **2021**, *21*, 1469. [CrossRef]
22. Hadad, O.; Bakalo, R.; Ben-Ari, R.; Hashoul, S.; Amit, G. Classification of Breast Lesions Using Cross-Modal Deep Learning. In Proceedings of the 2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017), Melbourne, VIC, Australia, 18–21 April 2017; IEEE: Piscataway, NJ, USA; pp. 109–112.
23. Shen, X.; Stamos, I. SimCrossTrans: A Simple Cross-Modality Transfer Learning for Object Detection with ConvNets or Vision Transformers. *arXiv* **2022**, arXiv:2203.10456.
24. Ahmed, S.M.; Lohit, S.; Peng, K.-C.; Jones, M.J.; Roy-Chowdhury, A.K. Cross-Modal Knowledge Transfer Without Task-Relevant Source Data. In *Computer Vision–ECCV 2022: Proceedings of the 17th European Conference, Tel Aviv, Israel, 23–27 October 2022, Proceedings, Part XXXIV*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 111–127.
25. Du, S.; Wang, Y.; Huang, X.; Zhao, R.-W.; Zhang, X.; Feng, R.; Shen, Q.; Zhang, J.Q. Chest X-ray Quality Assessment Method with Medical Domain Knowledge Fusion. *IEEE Access* **2023**, *11*, 22904–22916. [CrossRef]
26. Socher, R.; Ganjoo, M.; Manning, C.D.; Ng, A. Zero-Shot Learning through Cross-Modal Transfer. *Adv. Neural Inf. Process. Syst.* **2013**, *26*.
27. Chen, S.; Guhur, P.-L.; Schmid, C.; Laptev, I. History Aware Multimodal Transformer for Vision-and-Language Navigation. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 5834–5847.
28. Salin, E.; Farah, B.; Ayache, S.; Favre, B. Are Vision-Language Transformers Learning Multimodal Representations? A Probing Perspective. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 22 February–1 March 2022; Volume 36, pp. 11248–11257.
29. Li, Y.; Quan, R.; Zhu, L.; Yang, Y. Efficient Multimodal Fusion via Interactive Prompting. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 17–21 June 2023; pp. 2604–2613.
30. Srinivasan, T.; Chang, T.-Y.; Pinto Alva, L.; Chochlakis, G.; Rostami, M.; Thomason, J. Climb: A Continual Learning Benchmark for Vision-and-Language Tasks. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 29440–29453.
31. Falco, P.; Lu, S.; Natale, C.; Pirozzi, S.; Lee, D. A Transfer Learning Approach to Cross-Modal Object Recognition: From Visual Observation to Robotic Haptic Exploration. *IEEE Trans. Robot.* **2019**, *35*, 987–998. [CrossRef]
32. Lin, C.; Jiang, Y.; Cai, J.; Qu, L.; Haffari, G.; Yuan, Z. Multimodal Transformer with Variable-Length Memory for Vision-and-Language Navigation. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2022; pp. 380–397.
33. Koroteev, M. BERT: A Review of Applications in Natural Language Processing and Understanding. *arXiv* **2021**, arXiv:2103.11943.
34. Bao, H.; Dong, L.; Piao, S.; Wei, F. Beit: Bert Pre-Training of Image Transformers. *arXiv* **2021**, arXiv:2106.08254.
35. Yenter, A.; Verma, A. Deep CNN-LSTM with Combined Kernels from Multiple Branches for IMDb Review Sentiment Analysis. In Proceedings of the 2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), New York, NY, USA, 19–21 October 2017; pp. 540–546.
36. Ridnik, T.; Ben-Baruch, E.; Noy, A.; Zelnik-Manor, L. Imagenet-21k Pretraining for the Masses. *arXiv* **2021**, arXiv:2104.10972.
37. Yosinski, J.; Clune, J.; Bengio, Y.; Lipson, H. How Transferable Are Features in Deep Neural Networks? *Adv. Neural Inf. Process. Syst.* **2014**, *27*.
38. Liu, N.F.; Gardner, M.; Belinkov, Y.; Peters, M.E.; Smith, N.A. Linguistic Knowledge and Transferability of Contextual Representations. *arXiv* **2019**, arXiv:1903.08855.
39. Kirichenko, P.; Izmailov, P.; Wilson, A.G. Last Layer Re-Training Is Sufficient for Robustness to Spurious Correlations. *arXiv* **2022**, arXiv:2204.02937.
40. Kovaleva, O.; Romanov, A.; Rogers, A.; Rumshisky, A. Revealing the Dark Secrets of BERT. *arXiv* **2019**, arXiv:1908.08593.
41. Fushiki, T. Estimation of Prediction Error by Using K-Fold Cross-Validation. *Stat. Comput.* **2011**, *21*, 137–146. [CrossRef]
42. Goutte, C.; Gaussier, E. A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation. In *Advances in Information Retrieval: Proceedings of the 27th European Conference on IR Research, ECIR 2005, Santiago de Compostela, Spain, 21–23 March 2005, Proceedings 27*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 345–359.
43. Usmani, I.A.; Qadri, M.T.; Zia, R.; Alrayes, F.S.; Saidani, O.; Dashtipour, K. Interactive Effect of Learning Rate and Batch Size to Implement Transfer Learning for Brain Tumor Classification. *Electronics* **2023**, *12*, 964. [CrossRef]
44. Reddi, S.J.; Kale, S.; Kumar, S. On the Convergence of Adam and Beyond. *arXiv* **2019**, arXiv:1904.09237.

45. Niu, S.; Liu, Y.; Wang, J.; Song, H. A decade survey of transfer learning (2010–2020). *IEEE Trans. Artif. Intell.* **2020**, *1*, 151–166. [CrossRef]
46. Chui, K.T.; Gupta, B.B.; Chi, H.R.; Arya, V.; Alhalabi, W.; Ruiz, M.T.; Shen, C.W. Transfer learning-based multi-scale denoising convolutional neural network for prostate cancer detection. *Cancers* **2022**, *14*, 3687. [CrossRef] [PubMed]
47. Antol, S.; Agrawal, A.; Lu, J.; Mitchell, M.; Batra, D.; Zitnick, C.L.; Parikh, D. Vqa: Visual question answering. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015.