COMP3330 Group Project

Ismail Adebayo[1], Ethan Bird[1], Angelique Herfel[1],
Dexter Konijn[1] and Johanne Montano[1]

[1]The University of Newcastle, Australia
{c3328148, c3374129, c3309582, c3355260, c3336019}@uon.edu.au

Abstract. The Intel Image Classification and TweetTopic NLP datasets were used to examine the use of transfer learning to solve domain-specific classification tasks in both image and text-based problems. Various pre-trained convolutional neural networks (CNN) and Vision Transformer backbones were investigated for the Intel Image Classification dataset including ResNet18, ResNet50, VGG, EfficientNet and ViT. Varying hidden layers were added to these backbones, and then trained using the supplied train-validation split, with data preprocessing and augmentation applied. The best performing model was a transformer-based ViT model with an additional 20 neuron hidden layer and the final pretrained layer weights unlocked, which achieved an 85.8% top-1 accuracy on a custom test set. Several Bag-of-Words (BoW) and bidirectional long short term memory (LSTM) models were trained from scratch on the TweetTopic dataset. A BoW model with a hidden layer configuration of 100-100-100 trained using dropout regularisation produced the best candidate model with 69.96% accuracy on the provided test set. This was compared to models produced by fine tuning pretrained transformers including DistilBERT, BERT and RoBERTa using the HuggingFace API. These were trained with an adaptive learning rate scheduler and varying pretrained layers unfrozen, on using a larger dataset with concatenated augmentations. Using a fine tuned RoBERTa backbone, an accuracy of 86.26% was achieved, close to the maximum reported accuracy of 89% and significantly higher than what was able to be achieved using models trained from scratch.

Keywords: Image Classification, Transfer Learning, CNN, Vision Transformer, Bag of Words, Long Short Term Memory, Transformers

1 Intel Image Classification

The Intel Image Classification task contains a dataset of approximately 25,000 150x150 images originally compiled for the Intel Scene Classification Challenge[1]. The images are classified into six classes: buildings, forest, glacier, mountain, sea, and street. Both Convolutional Neural Network (CNN) and Vision Transformer models were investigated. Models were trained with transfer learning based on pretrained ResNet50, ResNet18, VGG and EfficientNet models and ViT.

1.1 Data Preprocessing

Inspection of the dataset revealed that image dimensions were sometimes inconsistent, however all images were resized to 224x224 using bilinear interpolation to match expected input dimensions of the base models. Additionally, manual data labelling and validation were necessary to ensure that poor data integrity did not undermine model performance and training. Some images in the glacier class were more suitably placed in the mountain class, and moved, while other irrelevant images were removed. This ambiguity is grounded in reality, as glaciers can sometimes be found on top of mountains, however it was crucial to establish a clear distinction between the two classes to ensure accurate training. Similarly, ambiguity was identified in the building and street classes (See Fig. 1) however this was not considered significant enough for relabelling.

Fig. 1. Images from each class of the Intel Image Classification dataset.

The supplied seg_pred dataset of approximately 3000 images was manually labelled using a helper script, which can be found in supplementary materials filename. This ensured enough test data to provide accurate results and test metrics, while ensuring there was no overlap with training data to eliminate the possibility of information leaks.

1.2 Training and Testing

Model creation was partially automated through the creation of an abstract CVModel class. This allowed various hyperparameters, including learning rate, batch size, data augmentations, model architectures and backbones, to be automatically tested. Multiple proposed hyperparameter combinations could be loaded into a script and be left to run overnight, the script would then export checkpoint models, training history, and various metrics which could be later reviewed. These scripts may be found in the supplementary material as models/cv_model.py, train_model.py and export.py.

To enhance the generalisation capability of the model, various data augmentation techniques were employed. These included colour jittering (adjusting brightness, hue, saturation, and contrast), Gaussian blur, random sharpness, and random horizontal flipping.

1.3 Max Pooling

The first consideration was whether to use max pooling, as it is common for many classification problems [2,3]. This was chosen as the first hyperparameter to run because this choice depends more on the nature of the task and data available than the model itself. This was tested on some rudimentary ResNet18 networks, and equivalent

networks with max pooling were found to produce a slightly worse result than networks without max pooling, so the idea was discarded early on to reduce the hyperparameter search scope.

1.4 Choosing a Convolutional Neural Network

The CNNs considered for the backbone were ResNet18, ResNet50, VGG and EfficientNet. For each prospective model the backbone was frozen and the head was replaced with a simple fully connected layer of size 6 (output size) to assess its suitability.

After the initial tests, the head of each model was replaced with 2-layers; one to introduce variation and another corresponding to the size of the output. All other hyperparameters were kept standard as a control, and the training runs were limited to 10 epochs. The purpose of this was to choose a model and architecture that was likely to succeed with further epochs and modifications to the training.

These prospective models were assessed using multiple metrics, including final accuracy, precision and recall on the prediction set, the accuracy graphed over time, their loss graphs and the confusion matrices. In particular, when two models had comparable final accuracies, the loss graphs and confusion matrices were examined. Loss graphs that had continued to trend downwards after 10 epochs were considered especially promising. Results of this testing are discussed in Section 1.5.

The first model backbones that were assessed were ResNet18 and ResNet50. These models were chosen as they showed good performance for their relative speed. Additionally, it was hypothesised they were less prone to overfitting due to ResNet's skip-connections, should they later be modified to unfreeze some layers [4].

VGG was also experimented with as it outperformed Resnet in some image classification tasks using pretrained weights [5]. This model was discarded because of its tendency to overfit quickly and show increasing loss values, despite its relatively high accuracy early in training, as well as producing inconsistent results between training and test sets.

Finally, variations of EfficientNet were also experimented with. It was observed that the larger model outperformed the smaller model, and that a larger batch size could help smooth out the learning peaks, showing a more uniform improvement over time. The consistency of results was promising, however training was slow and computationally expensive.

1.5 Adjusting the Training

After initial experimentation, all models were collected and training was done on similar architectures, with variations made to the training. In particular, the learning rate was reduced from the initial .001 to .0001, while the batch size was increased from 32 to 64 to improve the consistency of our results. Learning decay and Cosine Annealing were implemented to attempt to increase the area in which an effective model could be found by reducing the influence the training data had on the model in later epochs. It was found that as each run varied from the last this would need much more tuning than

was practical to be effective. During training, models were exported at five epoch intervals as well as upon completion of training, using the export.py script mentioned previously. This provided deeper insight into ideal training parameters, as performance metrics including confusion matrices could be evaluated over time.

Convolutional Neural Network Results. When comparing ResNet18 and ResNet50 backbones, ResNet18 was fast to train but did not perform as well as ResNet50. ResNet50 showing decent accuracies between 80 and 120 neurons. Both ResNet models had poor performance distinguishing the glacier and mountain classes (often misclassifying the majority of glacier images, despite high accuracy in all other classes), as well as difficulty with building and street classes. Training results, and some learning curves for ResNet models may be found in Appendix 3.1.

VGG was able to achieve high initial accuracies of ~80% even in the smallest VGG-10-6 model tested. The pretrained weights appeared to be adept at finding the subtle differences between the mountain and glacier classes, as anticipated by research. Significantly, the largest VGG-100-6 model achieved an 80% sensitivity in the glacier class, the best of any model (See Fig.7 in Appendix 3.2 for the confusion matrix). Unfortunately, this did not translate to consistent test-set performance, so other models were prioritized.

The EfficientNet backbone was found to be resistant to overfitting, even when tested up to 50 epochs (See Fig. 2). Unfortunately, while showing significant improvement early on, convergence was slow and was not able to achieve accuracies higher than ResNet50 in reasonable training times. Given more computational resources, it is likely that EfficientNet would have performed very well, as results were consistent and misclassifications were low, especially between glacier and mountain classes where other backbones struggled (See Fig. 14 in Appendix 3.3 for the confusion matrix).

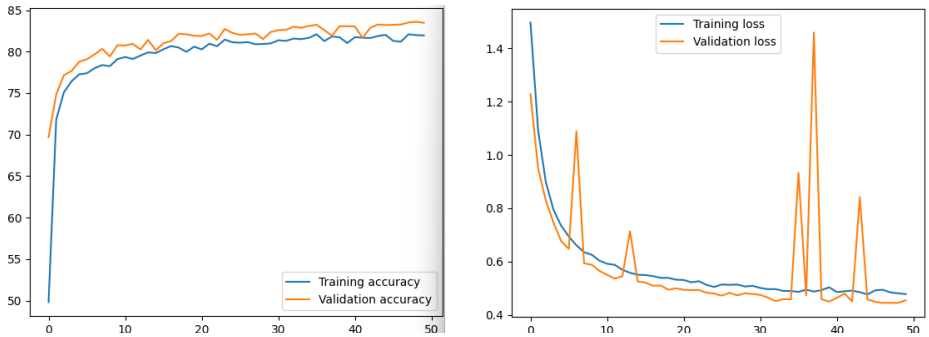

Fig. 2. Accuracy (left) and loss (right) plots per training epoch for the best EfficientNet model with a (-1280-6) hidden layer structure, trained for 50 epochs.

1.6 Vision Transformers

As most CNNs were found to have difficulty distinguishing between mountains and glaciers, transformers were explored as a potential solution to this problem. It was hypothesised that the transformer attention layers' ability to provide relational context could help the ANN distinguish between patterns which are very similar - such as small parts of rock or forest in mountains which otherwise may share both shape and colour with glaciers - where CNNs may struggle.

The vision transformer chosen was ViT. To train the vision transformer, the input images were again resized from (150, 150) to (224, 244). Initial testing showed promising results, with accuracies of around 75-80% in simpler architectures. Crucially, vision transformer models were much faster to train when compared to CNN models on a per-epoch basis.

Learning curves showed no signs of overfitting, such as diverging or increasing loss value, so the final layer of the model was unfrozen to improve results further. This was found to increase performance at the expense of training time, so despite downward trending loss curves, results did not improve significantly when training for a larger number of epochs. Appendix 3.4 documents results for ViT models, including loss curves and confusion matrixes.

Ultimately, vision transformers were found to resolve some ambiguity between mountains and glaciers, as well as streets and buildings, although this was only a minor improvement (See Fig. 3). The final best model was a Vision Transformer with a ViT*-20-6 hidden layer structure, with the final layer pretrained weights unfrozen. This model achieved a test accuracy of 85.8% on our custom 3000 image test set.

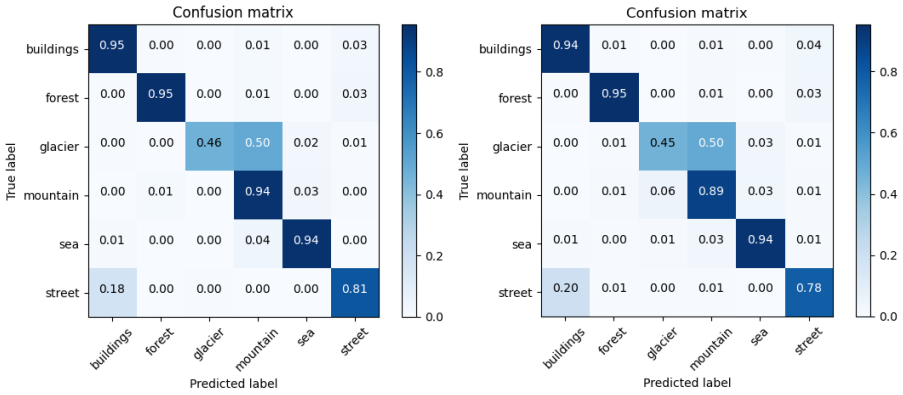

Fig. 3. Comparison of confusion matrices between the best performing ViT (left) and ResNet50 (right) models. ViT has slightly less misclassifications for glaciers, mountains and streets.

2 Tweet Topics Classifier

Various classifiers were investigated to categorise tweets from the CardiffNLP TweetTopic NLP dataset [6]. Topics categories included were arts_culture, business_entrepeneurs, pop_culture, daily_life, sports_&_gaming and science_&_technology. Training data from the provided datasets was split into training and validation samples with at 80-20 ratio, while all testing data was used as-is. Variations of the Bag of Words (BoW) and Long Short Term Memory (LSTM) models were created and trained. Results were compared to several variations of the BERT language model which were fine-tuned using the HuggingFace API. Training code for these models may be found in supplementary materials as files prefixed tweet_topics_.

2.1 Custom Neural Network Classifier

Bag of Words (BoW). The first BoW model was trained using a network with the hidden layers 256-128-64. The input vector was set as the vocabulary length of the dataset with 6 neurons as the output. This model was trained with a cross entropy loss function, a learning rate of 0.001, a batch size of 64 samples and a ReLU activation between each layer, with the ADAM optimiser for 20 epochs.

The model achieved an overall accuracy of 0.6658, with low precision and recall scores of 0.4701 and 0.4270 respectively. Strong overfitting was prevalent almost immediately, with validation loss sharply increasing after the third epoch (See learning curves in Appendix 3.5). Classification was particularly poor for arts_culture, business_entrepeneurs and science_&_technology classes, with the confusion matrix showing a near-uniform distribution in these domains (See Fig. 4).

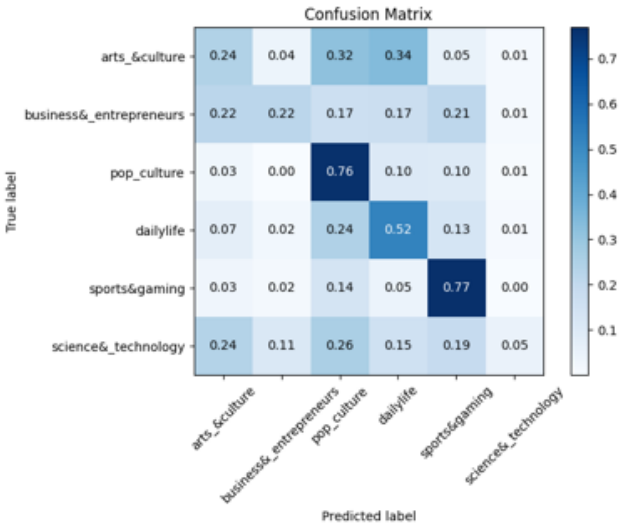

Fig. 4. Confusion matrix for a Bag-of-Words model with hidden layers 256-128-64 trained and tested on the TweetTopics NLP dataset.

Long Short Term Memory (LSTM). Initially, a bidirectional LSTM model was trained with the same hyperparameters as the previous BoW model, the embedding dimensions and hidden layers were both set to 64. The model was not found to overfit, but instead converged to a suboptimal solution (See Fig. 5) with an accuracy of 0.6584 and similarly low precision and recall of 0.4917 and 0.3902 as the previous BoW model. This caused more misclassifications in the arts_culture, business_entrepeneurs and science_&_technology classes (See Fig. 18 in Appendix 3.6).

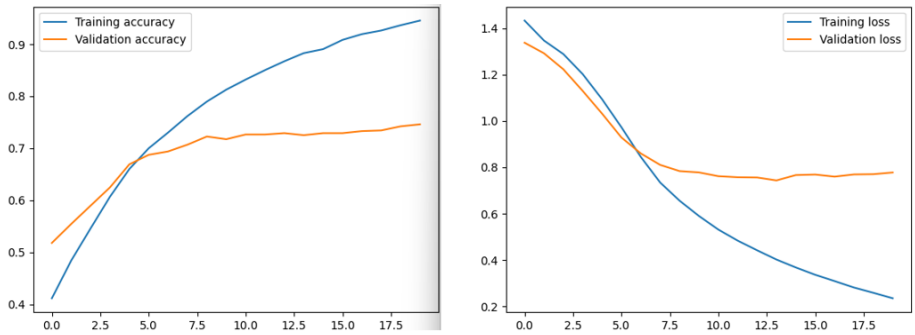

Fig. 5. Accuracy (left) and loss (right) plots for a bidirectional LSTM, showing early convergence to a suboptimal solution as the validation metrics plateu around the 8th epoch.

Several more LSTM models were trained with tweaking of the embedding dimensions and size of the hidden layers. Noticeably, larger embedding dimensions are able to capture more complex relationships and nuances between the words on average, and larger hidden layer sizes were found to produce more accurate inferences. In both cases, this traded fast training time for more accurate results. Even still, LSTM accuracy remained worse than the BoW model. It seemed that the nature of topic classification was better suited to models based on word presence, rather than word order (something more relevant for LSTM models).

Larger BoW Models. A second BoW model was trained, with the learning rate decreased to 0.0001 and with a hidden size structure of 256-256-256. Training was done for 30 epochs, and while the model began to overfit after approximately the 10th epoch, the lower learning rate was found to increase the maximum validation accuracy to 0.6861, and precision and recall to 0.4718 and 0.4294 respectively. Learning curves and the confusion matrix for this model may be found in Appendix 3.7.

While BoW models performed better that LSTM models, they were found to be significantly more prone to overfitting. The small size of the dataset was a contributing factor which made this difficult, however a final BoW model was trained with dropout to attempt to mitigate the chance of early overfitting during the training process. The model was trained with a smaller hidden layer configuration of 100-100-100 and a learning rate of 0.0001 for 35 epochs, with other hyperparameter staying the same.

While this did not eliminate overfitting, the loss curve exhibited a significant decrease in gradient (See Fig. 6), and ultimately this produced the best model with an accuracy of 0.6996. Although precision and recall were lower than most other models at 0.4561 and 0.3985 respectively, the use of dropout effectively limited overfitting and hence increased the accuracy of test set performance.

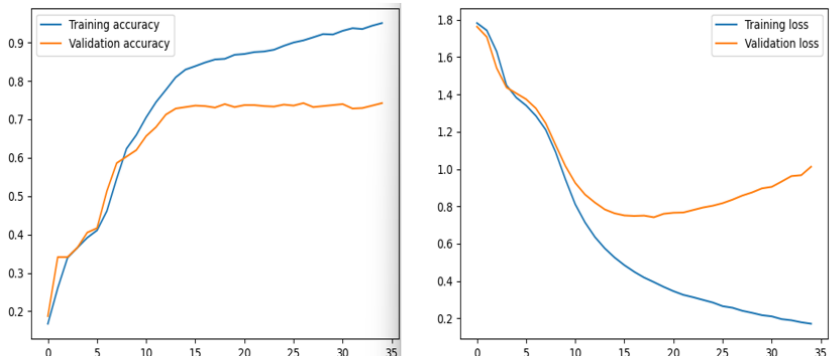

Fig. 6. Accuracy (left) and loss (right) plots for a BoW model with a 100-100-100 hidden layer configuration and learning rate of 0.0001.

Both BoW and LSTM models consistently were not able to establish a classifier that was sufficiently robust. All models consistently misclassified arts_culture as daily_life or pop_culture. Both business_entrepeneurs and business_entrepeneurs were consistently overclassified among all. This could be attributed to the uniqueness of the words and diverse vocabulary [6] which is inherently the nature of technical topics. All models consistently confuse daily_life with pop_culture, however this is consistent with the overlap of words that exist in the samples. All models reliably classified sports_&_gaming and pop_culture classes correctly, which is likely similarly due to the unique vocabulary.

2.2 Finetuning a Transformer

Approach. Variations of the BERT language model were trained using the HuggingFace python training library, making use of the adaptive learning rate scheduler. Models were chosen by size in ascending order, to minimise training time, using pretrained weights, with all layers unfrozen to allow for fine tuning. To better mitigate overfitting, the python package `niacin` was used for data augmentation to increase training sample size and model generalisation. Transforms applies were: random synonyms, hyponyms, misspellings, swapped words, contractions and misplaced whitespace. Transforms were applied to each sample with random probability and a low magnitude of 10 to ensure that the dataset was not altered severely so as to make it unfit for training. Due to the low probability of each augmentation, checks were run to see if the previous sample was the same as the new sample, and only sufficiently augmented data was concatenated to the raw dataset.

DistilBERT. The first model trained used DistilBERT, a model which claims to retain 97% of BERTs language understanding abilities while being 60% faster to train [7]. The model was trained for 10 epochs with a starting learning rate of 0.001 and a weight decay of 0.01 applied with 16 batches for both training and validation. Fine-tuned DistilBERT was found to immediately perform significantly better than the models trained from scratch. Test accuracy was 0.8390, with a precision and recall of 0.8444 and 0.8391, however evaluation metrics showed an increasing loss, while accuracy remained constant at approximately 0.87 (See Appendix 3.8). Significantly, DistilBERT had much more accurate classifications in the in the arts_culture, business_entrepeneurs and science_&_technology classes, where all BoW and LSTM models struggled significantly (See Fig. 7).

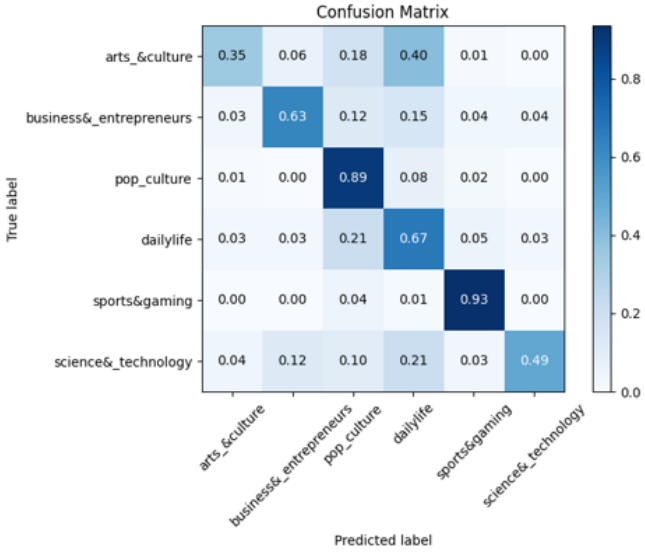

Fig. 7. Confusion matrix for a fine-tuned DistilBERT-based model trained and tested on the TweetTopics NLP dataset. Note the strong diagonal compared to earlier models tested, indicating high accuracy across almost all classes.

BERT. The base BERT model was chosen next and trained with an increased weight decay of 0.05 and starting learning rate of 0.0001. After a total of 10 epochs, the model had an higher test accuracy of 0.8503, with similar precision and recall of 0.8513 and 0.8501. Overfitting was still significant during the training process, and a constant increase in validation loss was observed after the first epoch (See Appendix 3.9 for training curves). Despite showing signs of overfitting, test metrics show better performance than the DistilBERT model. Comparing validation loss plots of the fine-tuned BERT and DistilBERT models shows increased weight decay helped to reduce overfitting as the rate of increase of the loss across the same number of epochs decreased significantly, resulting to a better performance on the test set. The confusion matrix for the test set may be seen in Appendix 3.9.

RoBERTa. The final model which was investigated for fine tuning was RoBERTa, and was expected to perform significantly better, as RoBERTa was trained on more data for a longer period of time compared to BERT [8]. Aditionally, the model uses dynamic masking to mask and replace tokens over each training epoch leading to better generalisation, and removed next sentence prediction to focus solely on the masked language modelling task which helps better classify text in the dataset [8].

For training, weight decay was increased to 0.05 and learning rate set to 0.0001. Batch size was dropped to 8 to reduce device strain, and all but the last few layers were unfrozen to help generalisation. Overall, this produced the best performing test metrics of all models evaluated, with an accuracy of 0.8626 and precision and recall of 0.8710 and 0.8627. Validation loss curves indicated overfitting (See Fig. 8) however the rate of increase was the lowest of all trained models. The confusion matrix (see Appendix 3.10) showed strong classifier performance for all classes, with the exception of arts_&_culture which was predominantly misclassified as daily_life.

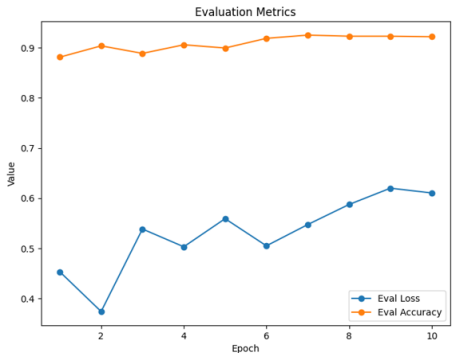

Fig. 8. Learning curved for a finetuned RoBERTa model trained and tested on the TweetTopics NLP dataset. Note the lower increase in loss as training progressed, indicating less overfitting.

2.3 NLP Findings

In general, pretrained transformer models performed significantly better compared to LSTM and BoW models that were trained from scratch. LSTM and BoW models show convergence and can easily avoid overfitting even with the small dataset, but their performance is suboptimal compared to the transformers. The pretrained transformers have been trained on a large volume of data hence have a wide vocabulary and understanding that is desirable for a text classification task.

Furthermore, transformers' attention layers help to contextualise data better, resulting in more accurate inferences, rather than just simply taking into account word order (LSTM) or only taking word count and frequency as a variable (BoW). One caveat is that bigger models are more hungry for data, making them more prone to overfitting. Smart utilisation of data pre-processing techniques such as normalisation and data augmentation and regularisation techniques like weight decay, learning rate scheduling, early stopping criteria, and unfreezing only later layers lead to better generalisation and more robust models that can make more accurate and better inferences.

References

1. Practice problem: Intel scene classification challenge, https://datahack.analyticsvidhya.com/contest/practice-problem-intel-scene-classification-challe/, accessed 02 May 2023.
2. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. 2017 IEEE International Conference on Computer Vision (ICCV). (2017).
3. Xie, S., Girshick, R., Dollar, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2017).
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2016).
5. Samhan, L., Alfarra, A., Abu-Naser, S., Amassi, I.: Classification of Alzheimer's Disease using Convolutional Neural Networks. International Journal of Academic Information Systems Research. (2022).
6. Antypas, D., Ushio, A., Camacho-Collados, J., Neves, L., Silva, V., Barbieri, F.: Twitter Topic Classification. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2209.09824 (2022)
7. Introduction to DistilBERT, https://www.analyticsvidhya.com/blog/2022/11/introduction-to-distilbert-in-student-model/, accessed 18 May 2023
8. Liu, Y., Ott, M., Goyal N., Du J., Joshi M., Chen D., Levy O., Lewis M., Zettlemoyer L., Stoyanov V.: RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv (Cornell University). https://doi.org/10.48550/arxiv.1907.11692

3 Appendix

3.1 Initial Testing of ResNet18 and ResNet50 Backbones

The following tests were done with a learning rate of 0.001 and batch size of 32, with a cross entropy loss function.

Table 1. Test accuracies for initial testing of various ResNet18-based architectures

Model Architecture	Test Accuracy	Epochs
ResNet18-10-6	81.63%	10
ResNet18-20-6	78.80%	10
ResNet18-30-6	82.10%	10
ResNet18-40-6	78.73%	10
ResNet18-50-6	78.67%	10
ResNet18-60-6	80.40%	10

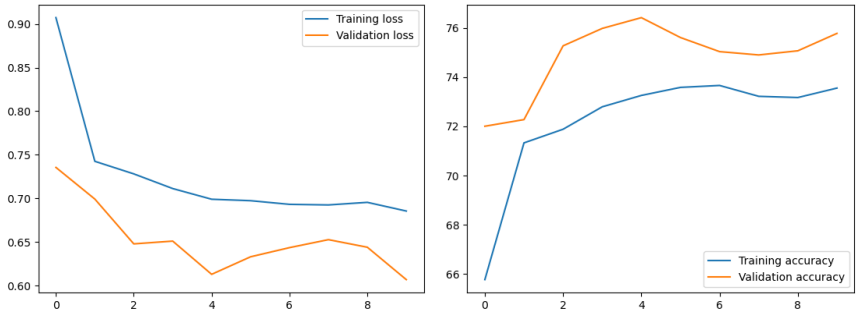

Fig. 9. Loss and accuracy curves for a ResNet18-based model tested with hidden later size (-50-6) displaying a downward trending loss even after 10 training epochs.

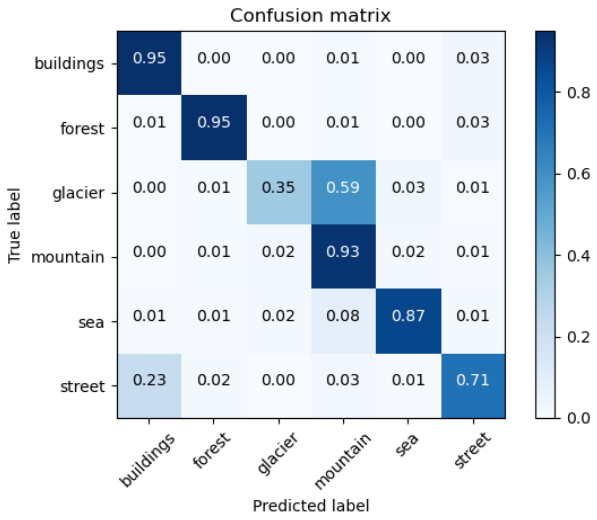

Fig. 10. Confusion matrix for the ResNet18-50-6 model from Fig. 6 above. Note the significant rate of misclassifications of glaciers and mountains.

Table 2. Test accuracies for initial testing of various ResNet50-based architectures. *Indicates the last layer of ResNet50 had weights unfrozen

Model Architecture	Test Accuracy	Epochs
ResNet50-6	78.67%	10
ResNet50-100-6	80.97%	10
Resnet50-150-6	82.73%	10
Resnet50-80-6	84.73%	10
Resnet50-80-6	82.37%	20
Resnet50*-80-6	84.30%	14

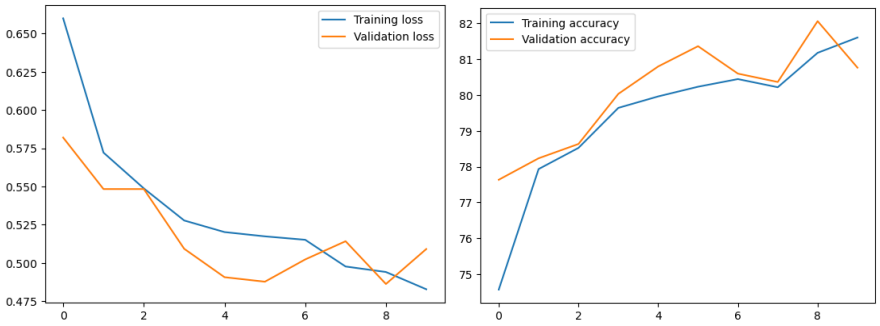

Fig. 11. Loss and accuracy curves for the best performing ResNet50-based model. After the fifth epoch validation loss stopped decreasing and overfitting began.

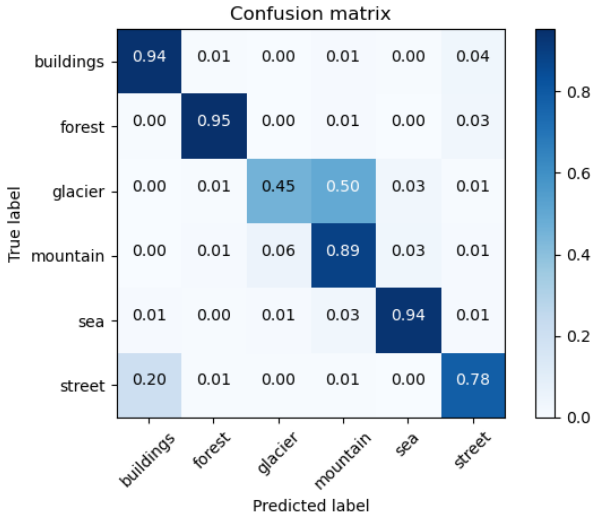

Fig. 12. Confusion matrix for the ResNet50 model mentioned above. Misclassifications were lower for mountains and glaciers compared to ResNet18 (See Fig. 8), despite still being high.

3.2 VGG Initial Testing Results

Table 3. Initial testing results of VGG-based architectures on the Intel Scene Classification task

Model	Accuracy	Epochs
VGG-10-6	81.2	15
VGG-30-6	78.73	15
VGG-100-6	81.8	10

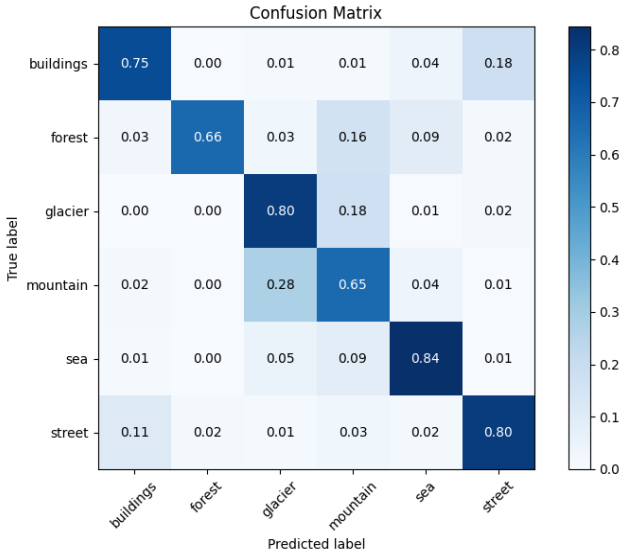

Fig. 13. Confusion matrix for the best model achieved for a VGG-based model. This model had an added hidden layer structure of (-100-6), corresponding to the last line in Table 4 above.

Fig. 14. Loss and accuracy plots for the training of the above VGG model (See Fig. 7). Validation and test accuracy was significantly different in this model, for reasons not understood.

3.3 EfficientNet Initial Testing Results

EfficientNet models were trained with a learning rate of 0.0001 and batch size of 64.

Table 4. Initial testing results of EfficientNetV2-based architectures on the Intel Scene Classification task.

Model	Accuracy	Epochs
EfficientNetb2-500-6	78.7%	15
EfficientNetb2-1280-6	81.3%	15

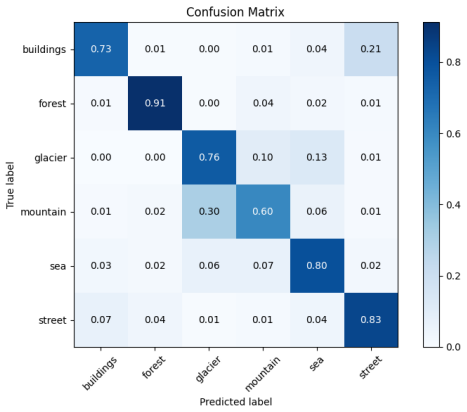

Fig. 15. Confusion matric for the EfficientNet_v2_medium (-1280-6) model mentioned above. While performance is not outstanding, results are consistent between classes, and misclassification is low compared to other models.

3.4 Vision Transformer Results

ViT-based models were trained with a learning rate of 0.0001 and batch size of 64. Where a model is annotated as (Unfrozen) this indicates that the final layer (prior to additional hidden layers being added) of the ViT model has been unfrozen.

Table 5. Test accuracies for transformer models tested with a ViT backbone. *Indicates the final layer of ViT had weights unfrozen during training.

Model	Test Accuracy	Epochs
ViT-30-6	81.8%	10
ViT*-50-6	85.2%	18
ViT*-20-6	85.8%	10

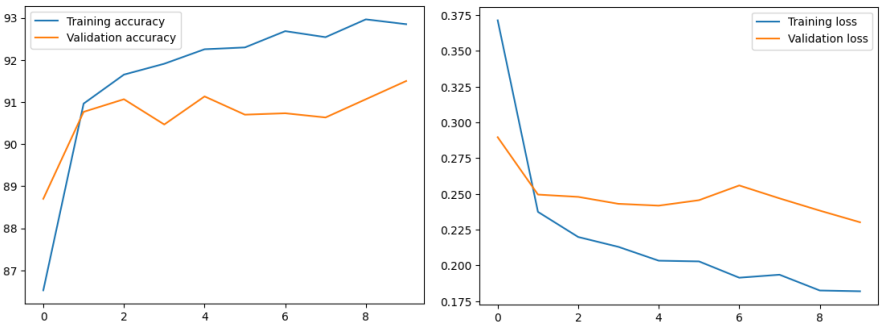

Fig. **16.** Accuracy and loss curves for the best performing ViT model with a ViT*-20-6 architecture (See Table 3). Note the validation loss reached a plateau without overfitting.

3.5 Initial BoW Model Results

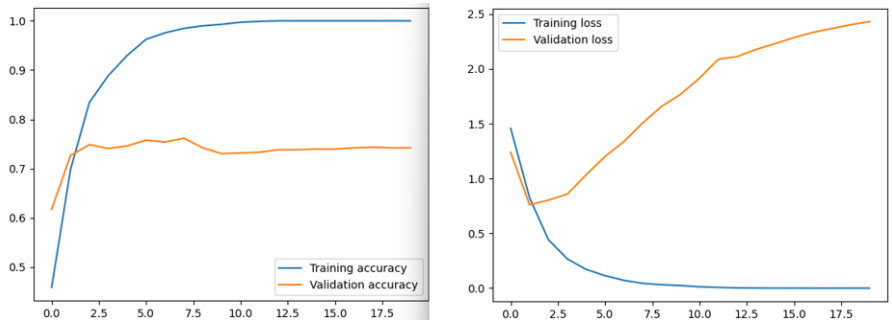

Fig. 17. Accuracy (left) and loss (right) plots per training epoch. Note the significant overfitting as validation loss sharply increases.

3.6 LSTM Model Results

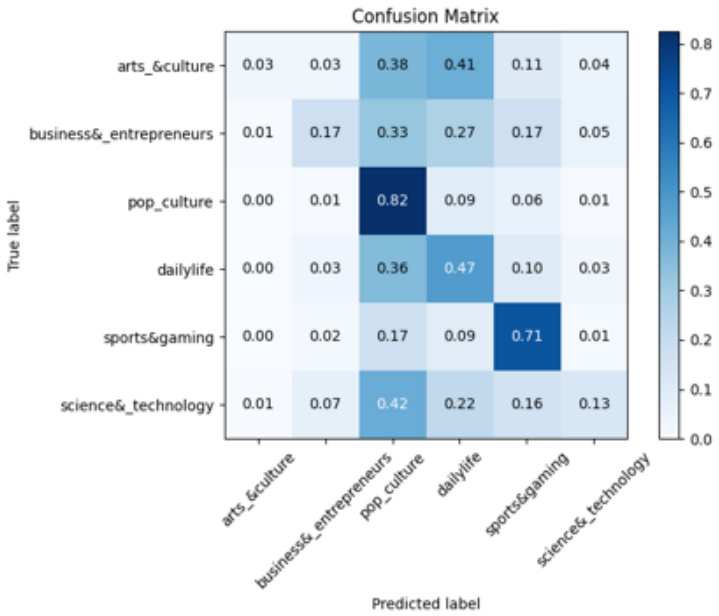

Fig. 18. Confusion matrix for the first bidirectional LSTM model trained and tested on the TweetTopics NLP dataset.Larger BoW Model Results

3.7 Larger BoW Model Results

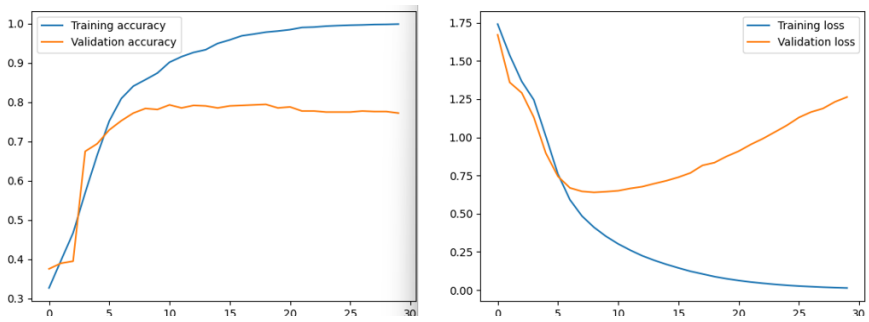

Fig. 19. Accuracy and loss plots per training epoch for a second BoW model, with the learning rate decreased to 0.0001 and with a hidden size structure of 256-256-256

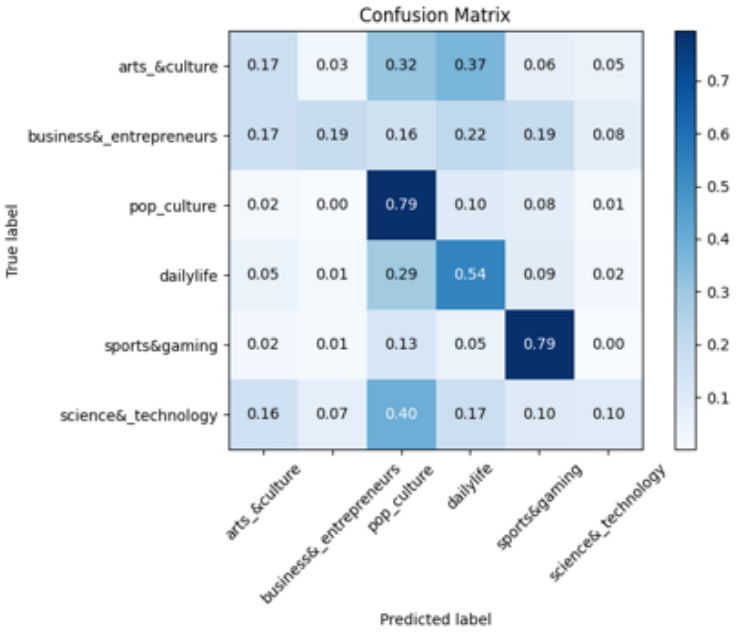

Fig. 20. Confusion matrix for a second BoW model, with the learning rate decreased to 0.0001 and with a hidden size structure of 256-256-256

3.8 DistilBERT Finetuning Results

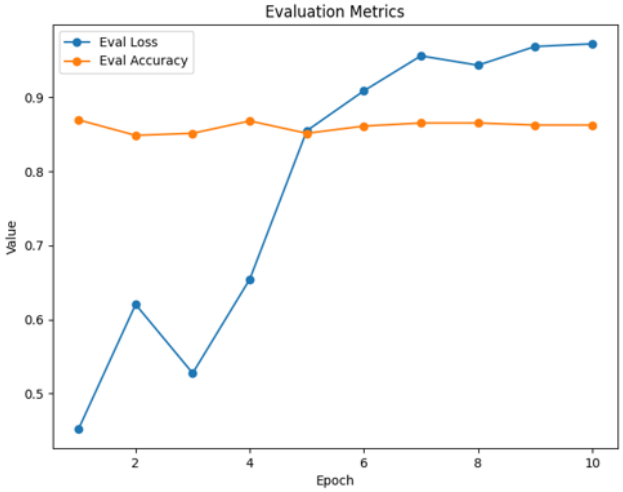

Fig. 21. Learning curves for a finetuned DistilBERT-based model

3.9 BERT Finetuning Results

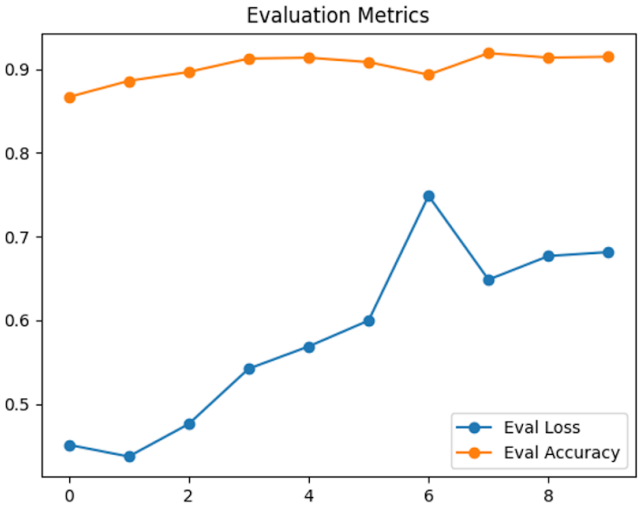

Fig. 22. Learning curves for a finetuned BERT model.

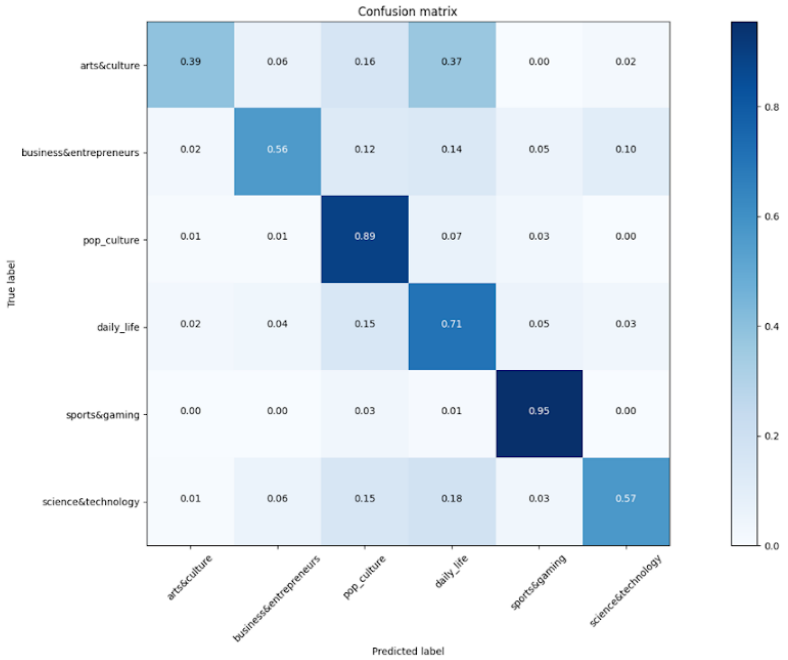

Fig. 23. Confusion matrix for a finetuned BERT model

3.10 RoBERTa Finetuning Results

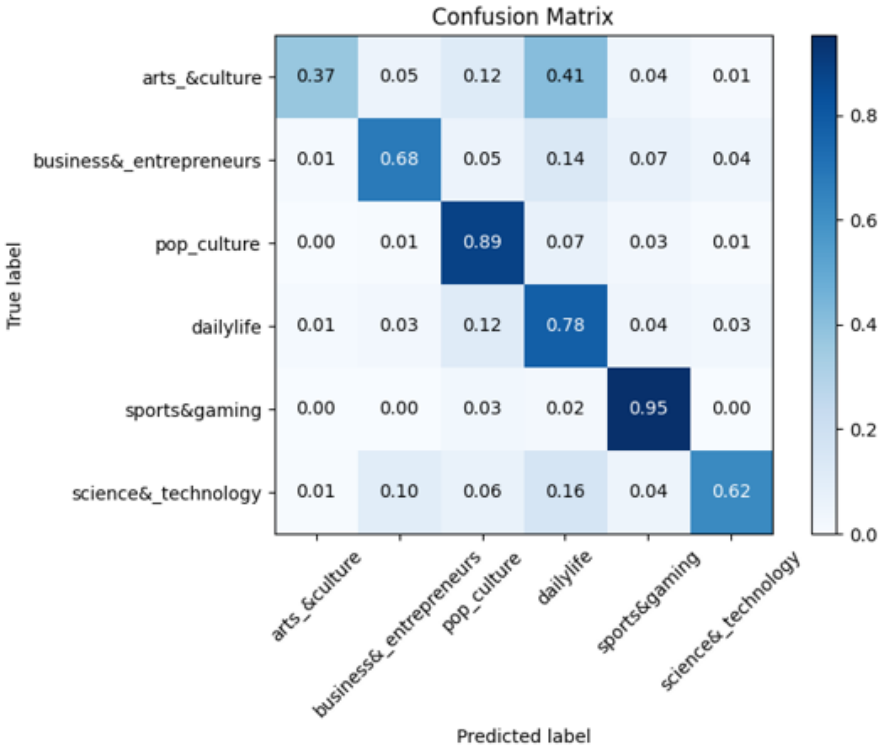

Fig. 24. Confusion matrix for a fine-tuned RoBERTa based model trained and tested on the TweetTopics NLP dataset.