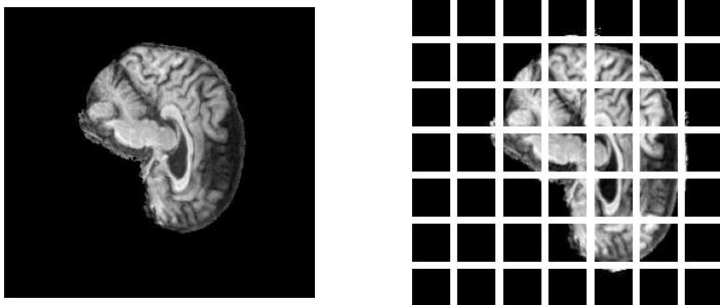


Classification of Alzhimers Disease

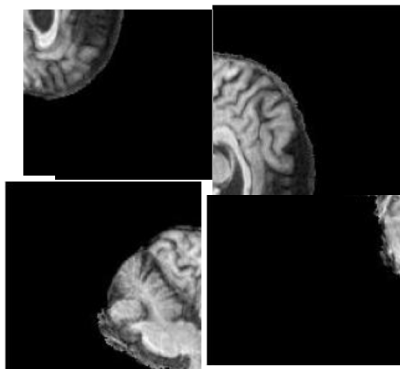
This is an example of a Vision Transformer used to classify a person who has AD (Alzhimers Disease) or NC (Normal). Vision transformers are an image classification model that utilises a transformer type architecture on patches of an image. This is achieved in 4 main steps [2].

How the model works

Step 1: Creating image patches



In this step images that are created into patches and flattened in a linear sequence (Positional Embedding). By doing so the transformer is able to remember the ordering of each patch. This is important because an image could have the same patches but they aren't in the right order. An example of this is shown below.



Step 2: Patch Encoding

Patch encoding can be done in various different ways depending on the architecture of the transformer. In this case a multilayer perceptron is used along with a sigmoid classification layer at the end. These help the vision transformer do the image classification (AD or NC).

Step 3: Training

In this step the model created from the mlp and transformer layers is trained till a satisfactory va_accuracy is reached. A AdamW optimizer alongside with a BinaryCrossEntropy loss function is used in this process to improve the model.

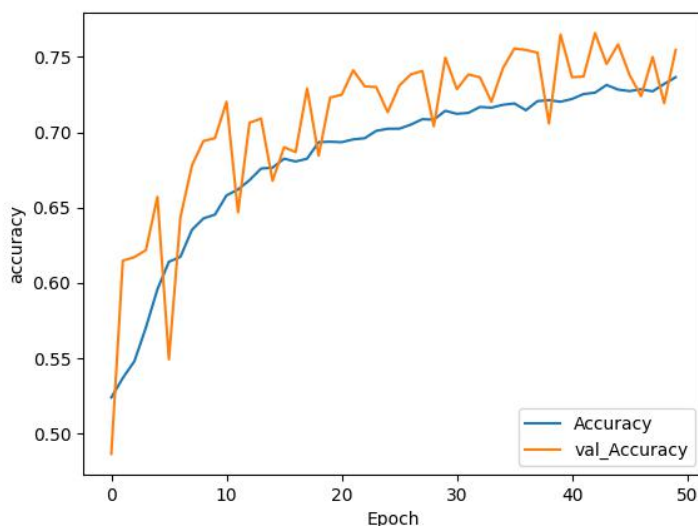
Step 4: Testing

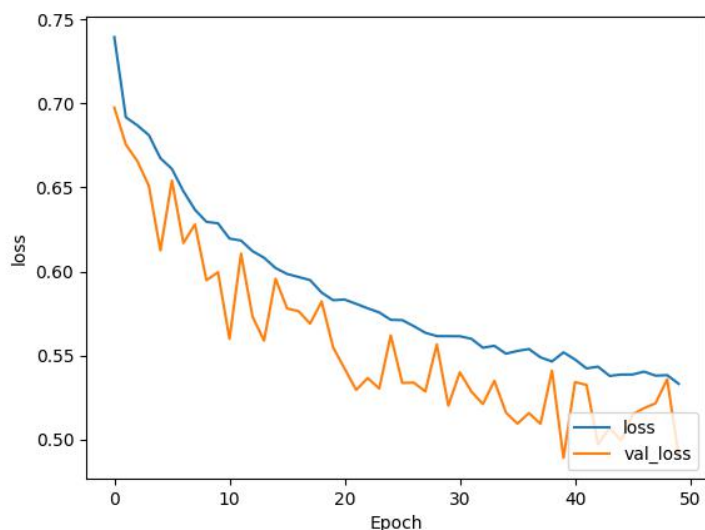
In this final step the data collected from the training model is plotted into loss and accuracy graphs. In addition, the model is finally tested on testing data which outputs the final accuracy of it

How the model performs

Currently, the model performs approximately at 65% accuracy with a loss and accuracy graph as shown below.

```
Epoch 46/50
303/303 [=====] - 17s 56ms/step - loss: 0.5427 - Accuracy: 0.7246 - val_loss: 0.4749 - val_Accuracy: 0.7807
Epoch 47/50
303/303 [=====] - 17s 56ms/step - loss: 0.5390 - Accuracy: 0.7299 - val_loss: 0.4810 - val_Accuracy: 0.7770
Epoch 48/50
303/303 [=====] - 17s 56ms/step - loss: 0.5426 - Accuracy: 0.7249 - val_loss: 0.4792 - val_Accuracy: 0.7681
Epoch 49/50
303/303 [=====] - 17s 56ms/step - loss: 0.5367 - Accuracy: 0.7279 - val_loss: 0.4911 - val_Accuracy: 0.7663
Epoch 50/50
303/303 [=====] - 17s 56ms/step - loss: 0.5360 - Accuracy: 0.7298 - val_loss: 0.4804 - val_Accuracy: 0.7756
282/282 - 4s - loss: 0.6553 - Accuracy: 0.6543
test_loss: 0.6553295254707336
test_acc: 0.6543333530426025
is_anything_else_being_returned: []
```





While such a low accuracy is not ideal, the following methods were done in an attempt to improve the accuracy

Method 1: Increasing Epochs and reducing image size

The cluster provided to train has a limited gpu allocation. Therefore, if the training model used surpasses that limit, the training will immediately halt. This was one of the biggest limitations of the task. As such to avoid exceeding this limit one strategy used was to decrease the image size. By doing so the quality of the image is lowered thereby making it easier for it to train the model. However, by doing so one limitation of this is more epochs are required to get higher val_accuracy's. This strategy was implemented using two different loss functions SparseCategoricalCrossentropy and BinaryCrossentropy. By doing so I aimed to figure out which loss function worked best. Overall, no drastic differences were observed both loss functions outputted an accuracy of 63% +/- 1%.

Method 2: Decreasing Epochs and increasing image size

Similar to the method earlier in this method the image size is increased by doing so the model is able to capture more detail about each classification. However, the limitation of using this method is that the number of epochs the cluster can run is reduced (due to the higher image resolution). This strategy was also implemented using two different loss functions SparseCategoricalCrossentropy and BinaryCrossentropy. Overall, no drastic differences were observed both loss functions outputted an accuracy of 63% +/- 1%.

Method 3: Normalizing the data

In this method each image that is passed through the transformer is normalised. By doing so, I aimed to increase the accuracy of the model [4]. By implementing this technique no observable changes in accuracy was

made. This strategy was also implemented using two different loss functions SparseCategoricalCrossentropy and BinaryCrossentropy. The final test accuracy after doing so was 63% +- 1% in both cases.

Method 4: Adding a softmax/sigmoid activation in the last layer

In this method a softmax layer was added in as it was what Source [1] recommended using for the final layer. By doing so the model's accuracy hit a maximum of 48%. Therefore, this layer was not used in the final model. A sigmoid layer as suggested by a tutor was later used and did improve the accuracy by 2% resulting in an average accuracy of 65%.

Method 5: Adjusting the parameters of the vision transformer:

The vision transformer has other parameters that can be used to fine tune the model such as

- Patch size
- Projection dimensions
- Number of heads
- Transformer units

According to Source [2] by increasing these values it is possible to increase the accuracy of the transformer. Therefore, they were increased/decreased in different variations. However, no combination that I have tried so far has resulted in a better testing accuracy.

Method 6: Adding a convolution layer in the multi layer perceptron:

This method did not have any reasoning behind it. It was simply a random idea that I decided to implement as a last resort. However, no improvements in the model were observed. As such I did not use it.

Preprocessing and testing/training/validation dataset

testing/training/validation

For the testing and validation a 90:10 split was used as it was a commonly used split in most datasets for machine learning [5]. The testing set was provided separately in the data set therefore, no additional splits were required for it.

Pre-processing

To pre-process the data a data augmentation method was used as it was present in source [2] and also recommended by sources online such as [6] and [7]. By implementing this method variations of the images passed through are created. This therefore, increases the variety of different images (useful when there is not a

lot of training data given). In this preprocessing model the images undergo a normalaization, Resizing, RandomFlip, RandomRotation, and RandomZoom.

While not a lot of research was done on this section from my part I believe that to improve the model further reasearch could be done in this space to potentially improve the accuracy of the vision transformer.

Dependencies

To run this model a few dependencies need to be used namely

- tensorflow
- numpy
- os
- cv2 (install opencv)
- random
- matplotlib
- keras/keras.layers
- matplotlib.pyplot
- tensorflow_addons

Further to run the model the ADNI data set needs to be downloaded from <https://adni.loni.usc.edu/>. After downloading the dataset, load it into the solution folder and ensure that the path DATADIR in dataset.py is updated for both training and testing functions.

After doing so the command: `srun -p test -G1 python3 modules.py` can be used to run the model.

Sources

[1] <https://www.youtube.com/watch?v=j-3vuBynnOE>

[2] https://www.youtube.com/watch?v=i2_zJ0ANrw0

[3] https://keras.io/examples/vision/image_classification_with_vision_transformer/

[4] <https://towardsdatascience.com/data-normalization-in-machine-learning-395fdec69d02#:~:text=The%20short%20answer%20is%20%E2%80%94%20it,be%20cause%20they%20are%20big%20numbers.>

[5] <https://www.v7labs.com/blog/train-validation-test-set#:~:text=In%20general%2C%20putting%2080%25%20of,dimension%20of%20the%20data%2C%20etc.>

[6] <https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/>

[7] <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0197-0>

Note for tutor

For my git logs I have had the problem of pushing my algorithm from the cluster. Therefore, I have instead for my solution created a local copy of it and created git logs that I backtracked based on my previous commits. As such most commits done are all done in the span of 2 days.

I have provided screenshots as proof for this

File Edit Selection View Go Run Terminal Help MySolution [SSH: rangpur.compute.eait.uq.edu.au] - Visual Studio Code

EXPLORER

OPEN EDITORS

MY SOLUTION [SSH: RANG...]

__pycache__

ADNI_AD_NC_2D

.gitignore

dataset.py

modules.py

OG.jpg

predict.py

README.MD

Test.jpg

train.py

utils.py

PROBLEMS OUTPUT PORTS JUPYTER DEBUG CONSOLE TERMINAL

git log

commit ea8edf7359b3c366a31c61cbca2e674d2dec1d50 (HEAD -> master)

Author: pranaypremdas <pranaypremdas@yahoo.com.au>

Date: Fri Oct 14 10:18:06 2022 +1000

git cache cleared

commit 7507aa890ca2d990851728f80561fb657a91b562

Author: pranaypremdas <pranaypremdas@yahoo.com.au>

Date: Fri Oct 14 10:12:48 2022 +1000

Untrack files in .gitignore

commit 217aa815da93da10c47c664dfef1df69742944cf

Author: pranaypremdas <pranaypremdas@yahoo.com.au>

Date: Fri Oct 14 10:01:01 2022 +1000

Creating pull request

commit 9f6d9f2c1f174b48a463b190804b1bc2bdca2c27

Author: pranaypremdas <pranaypremdas@yahoo.com.au>

Date: Fri Oct 14 09:27:17 2022 +1000

Reverted model to not have Conv2d layer as it did not improve it

commit 0ef7e1b9766c281894ae4a97c13005a065305669

Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>

Date: Fri Oct 14 08:43:23 2022 +1000

Increased patch size to 32, increased image size to 250, and decreased epochs to 50. Average accuracy is still 63%

commit 39dde4a5ecbf14622c5ac2177cddb082350b794cd

Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>

Date: Wed Oct 12 06:14:01 2022 +1000

Increased the num_heads, and projection_dim in the model. Current val_accuracy is still approximately 0.72

commit 1273d03fb0b89e705ce90880a65b12ff355de857

Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>

Date: Tue Oct 11 18:07:12 2022 +1000

Added new modifications to the number of heads, prejection_dim, and patch_size. Accuracy is yet to be tested. NOTE: more refere nces need to be added to each code block to avoid plagariism penalty.

commit f69964120e0e6a7aa6bcb37318ede128bc143c60

Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>

Date: Tue Oct 11 13:29:15 2022 +1000

Implimented new tweaks to the model

- increased projection dim factor
- increased patch size
- increased number of mlp heads to 2048, 1024, 512, 256

Accuracy increased to 0.72 on average

commit fa7f64f32e9c2ed2425af662e2a44a1203e52b2f

Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>

Date: Mon Oct 10 20:22:25 2022 +1000

Added training code for the transformer. Accuracy is currently 0.62 on average. Requires a better training model and more comme nting

commit 5e906bd56c76102e75918ca83e461539aa704e69

Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>

Date: Mon Oct 10 20:22:25 2022 +1000

SSH: rangpur.compute.eait.uq.edu.au master 01 121 0 0 0 Connected to Discord

File Edit Selection View Go Run Terminal Help MySolution [SSH: rangpur.compute.eait.uq.edu.au] - Visual Studio Code

EXPLORER

OPEN E...

MY SOLUTION [SSH: RANG...]

__pycache__

ADNI_AD_NC_2D

.gitignore

dataset.py

modules.py

OG.jpg

predict.py

README.MD

Test.jpg

train.py

utils.py

PROBLEMS OUTPUT PORTS JUPYTER DEBUG CONSOLE TERMINAL

commit 0ef7e1b9766c281894ae4a97c13005a065305669

Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>

Date: Fri Oct 14 08:43:23 2022 +1000

Increased patch size to 32, increased image size to 250, and decreased epochs to 50. Average accuracy is still 63%

commit 39dde4a5ecbf14622c5ac2177cddb082350b794cd

Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>

Date: Wed Oct 12 06:14:01 2022 +1000

Increased the num_heads, and projection_dim in the model. Current val_accuracy is still approximately 0.72

commit 1273d03fb0b89e705ce90880a65b12ff355de857

Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>

Date: Tue Oct 11 18:07:12 2022 +1000

Added new modifications to the number of heads, prejection_dim, and patch_size. Accuracy is yet to be tested. NOTE: more refere nces need to be added to each code block to avoid plagariism penalty.

commit f69964120e0e6a7aa6bcb37318ede128bc143c60

Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>

Date: Tue Oct 11 13:29:15 2022 +1000

Implimented new tweaks to the model

- increased projection dim factor
- increased patch size
- increased number of mlp heads to 2048, 1024, 512, 256

Accuracy increased to 0.72 on average

commit fa7f64f32e9c2ed2425af662e2a44a1203e52b2f

Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>

Date: Mon Oct 10 20:22:25 2022 +1000

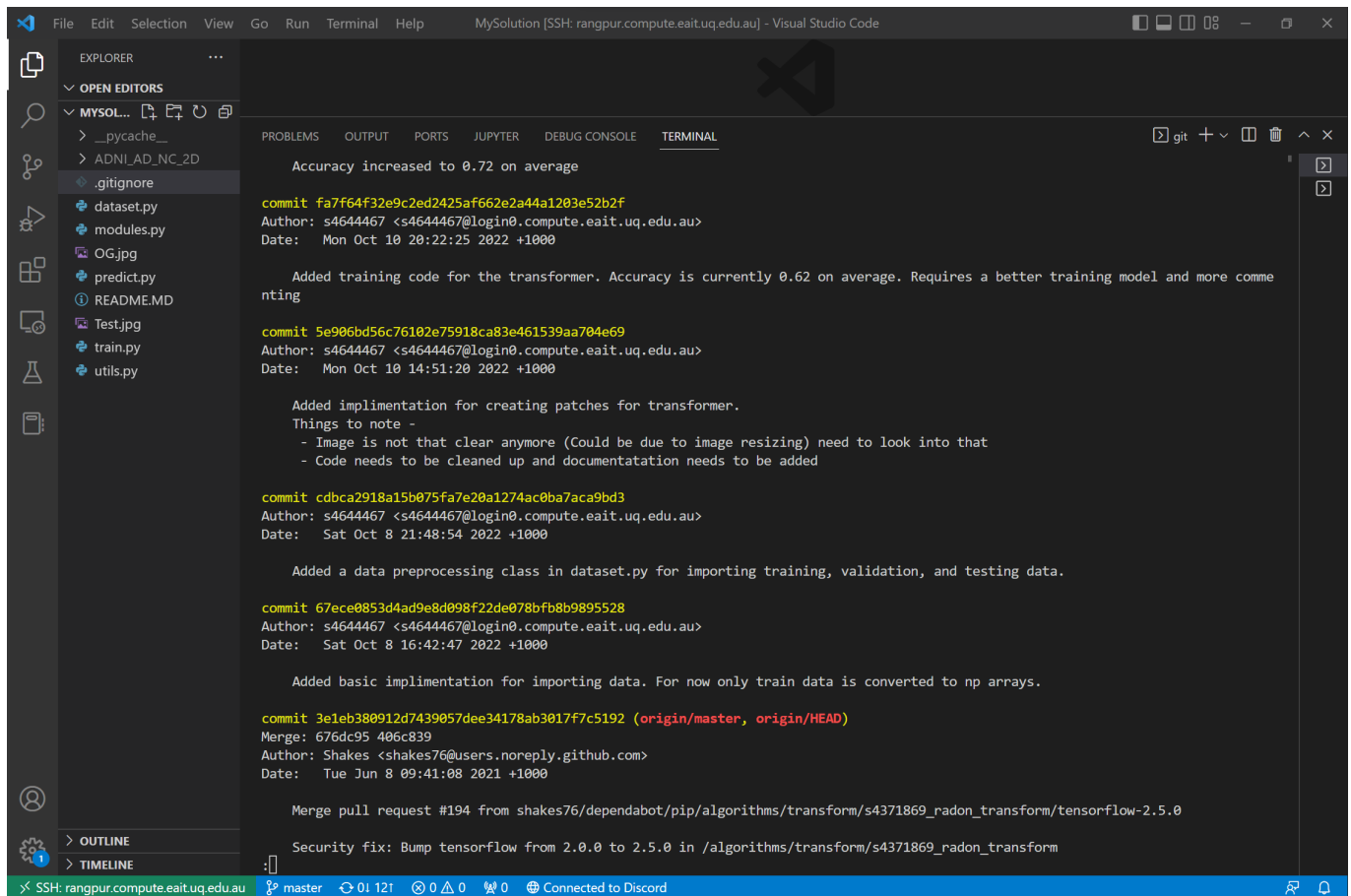
Added training code for the transformer. Accuracy is currently 0.62 on average. Requires a better training model and more comme nting

commit 5e906bd56c76102e75918ca83e461539aa704e69

Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>

Date: Mon Oct 10 20:22:25 2022 +1000

SSH: rangpur.compute.eait.uq.edu.au master 01 121 0 0 0 Connected to Discord



The screenshot shows the Visual Studio Code interface with a terminal window open. The terminal displays a series of git commit messages and status updates. The file explorer on the left shows a project structure with files like `dataset.py`, `modules.py`, `OG.jpg`, `predict.py`, `README.MD`, `Test.jpg`, `train.py`, and `utils.py`.

```
Accuracy increased to 0.72 on average

commit fa7f64f32e9c2ed2425af662e2a44a1203e52b2f
Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>
Date: Mon Oct 10 20:22:25 2022 +1000

    Added training code for the transformer. Accuracy is currently 0.62 on average. Requires a better training model and more commenting

commit 5e906bd56c76102e75918ca83e461539aa704e69
Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>
Date: Mon Oct 10 14:51:20 2022 +1000

    Added implimentation for creating patches for transformer.
    Things to note -
    - Image is not that clear anymore (Could be due to image resizing) need to look into that
    - Code needs to be cleaned up and documentatation needs to be added

commit cdbca2018a15b075fa7e20a1274ac0ba7aca9bd3
Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>
Date: Sat Oct 8 21:48:54 2022 +1000

    Added a data preprocessing class in dataset.py for importing training, validation, and testing data.

commit 67ece0853d4ad9e8d008f22de078bfb8b9895528
Author: s4644467 <s4644467@login0.compute.eait.uq.edu.au>
Date: Sat Oct 8 16:42:47 2022 +1000

    Added basic implimentation for importing data. For now only train data is converted to np arrays.

commit 3e1eb380912d7439057dee34178ab3017f7c5192 (origin/master, origin/HEAD)
Merge: 676dc95 406c839
Author: Shakes <shakes76@users.noreply.github.com>
Date: Tue Jun 8 09:41:08 2021 +1000

    Merge pull request #194 from shakes76/dependabot/pip/algorithms/transform/s4371869_radon_transform/tensorflow-2.5.0

Security fix: Bump tensorflow from 2.0.0 to 2.5.0 in /algorithms/transform/s4371869_radon_transform
```