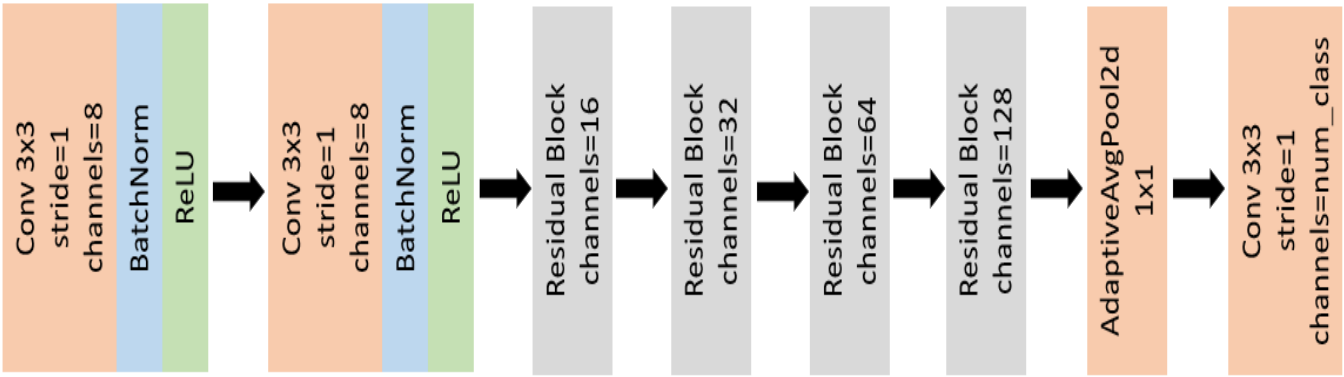
Task-1

In this task we will detect emotions from static facial images. More specfically, we will implement the Xception architecture for detecting emotions and understand its efficacy as an image recognition pipeline. We will make use of the Facial Expression Recognition (FER2013) dataset. The dataset consists of 35,685 examples of 48x48 pixel grayscale images of faces. Images are categorized based on the emotion shown in the facial expressions (happiness, neutral, sadness, anger, surprise, disgust, fear).

Donwload the code for this task. Package will be downloaded as `Task-1.zip`. Extract the folder to obtain the directory structure consisting of the following folders-

- `dataset`- The folder contains FER2013 dataset as a single `.csv` file.
- `src`- The folder contains `train.py`, `model.py` and `check.py`. We will use these files to implement and improve the Xception model.
- `test`- The folder contains a set of 4 test cases.
- `test2`- The folder contains a different set of 4 test cases.

Install the required libraries listed in `requirements.txt` using `pip install -r requirements.txt`.



1. Figure above presents the Xception model for emotion recognition on FER2013 dataset. Implement the figure in `model.py` using PyTorch. The model class along with the required blocks has already been implemented for you. Write your program in the `init` and `forward` functions of the class. **(1 pt)**
2. Complete the functions `plot_loss()` and `plot_acc()` in `train.py` for plotting loss and accuracy metrics. **(1 pt)**
3. Train the model for 20 epochs by running `train.py` file (`python train.py`). This should take approximately 16 minutes of wall-clock time on your local CPU machine. You will observe two kinds of metrics while training - public and private metrics. These are based on data samples which are made available to the model. Upon completion of training, report training accuracy, training loss, private test accuracy, private test loss, plot for loss curve and plot for accuracy curve. **(1 pt)**
4. We will now evaluate the model and gain intuition towards the representations learned by the separable convolutional layers of Xception architecture. The `test` folder contains 4 test images named after their true labels. Run `check.py` file (`python check.py`). Open and report `guided_gradcam.jpg` file. Observe the model predictions and activations corresponding to test images. Can you comment on the performance of the model? **(1 pt)**
5. Delete the contents of `test` folder and copy the contents of `test2` folder in `test` folder. Now run `check.py`. Open and report `guided_gradcam.jpg` file. What happened to the model predictions and

activations? Why? **(2 pts)**

6. Using **only one** of (a) *finetuning*, (b) *transfer-learning* or (c) *class-reweighting*, conduct a small experiment to address the limitations of the model observed above. Which method did you select? Why? **(2 pts)**
7. How does the selected method improve the model? Explain your approach and its limitations in detail. Note that the emphasis is not on the improvement of metrics but on implementing your ideas in order to address the limitations of the model. **(4 pts)**