

Understanding Clouds from Satellite Images



❖ **Course Details**

- Course Name: Topics in Data Analytics
- Course ID: CISC-839
- Course Instructor: Hazem Abbas

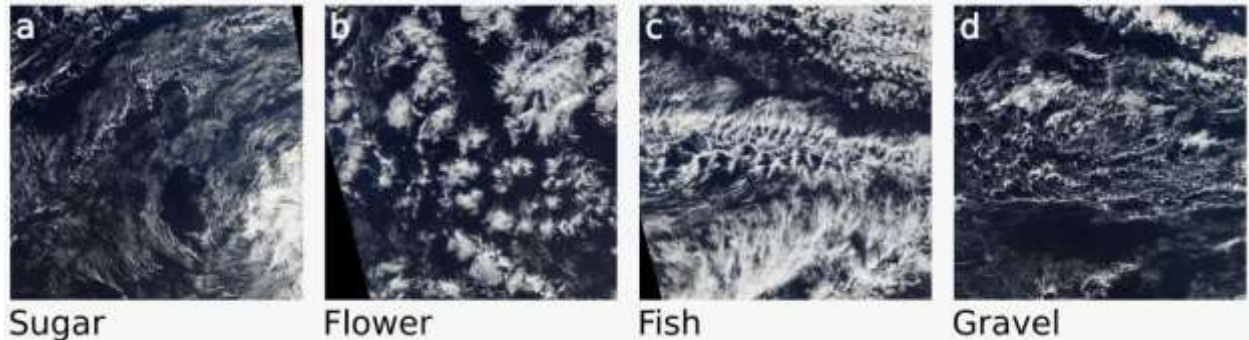
❖ **Group Members**

- Karthik Ranga Swamy: 20188884
- Abhishek Awasthi: 20143052
- Dhanush Reddy Nandyala: 20134832

❖ **PROBLEM DESCRIPTION & MOTIVATION**

Climate change has been at the top of our minds and there are many ways in which clouds can organize, but the boundaries between different forms of organization are not fully explained. This makes it challenging to build traditional rule-based algorithms to separate cloud features to better understand the clouds. Shallow clouds play a huge role in determining the Earth's climate and they are difficult to understand and to represent in climate models. In this Project we will be identifying regions in satellite images that contain certain cloud formations shown below, with label names: Fish, Flower, Gravel and Sugar. The segment for each cloud formation label for an image is encoded into a single row, even if there are several non-contiguous areas of the same formation in an image. Each image has at least one cloud formation, and can possibly contain up to all four. Stating the above we can consider this as a multiclass segmentation task which is

finding 4 different patterns in the images. Since, we make predictions for each pair of image and segmentation mask(label) separately, this could be treated as a 4 binary segmentation problem. In order to solve this task, we decided to implement two deep learning segmentation models Mask Region based Convolution neural network(Mask-RCNN) and UNET.



❖ CHALLENGES FACED:

- Selection of the Deep learning model was the biggest challenge we faced as none of us had any prior experience in image segmentation. We wanted to implement a powerful model with good documentation and also with a reasonable training time but most of the state-of-the-art models required a lot of training time and had poor documentation as a result we had to compromise among performance or training time in selecting the segmentation model.
- Lack of computation power was another challenge as training images are computationally expensive and require a large amount of time. (On an average it took 90 min for one epoch for Mask-RCNN)
- Data structuring for selected ANN Models and corresponding Backbones proved to be a difficult task as each model and implemented backbone requires custom data generator classes.
- One of the biggest challenges was to train and predict masks which were overlapping each other shown in the figure-1 and masks which contained a lot of background noise. In instance segmentation each pixel is classified and given a class and when masks overlap each other it is very hard for the model to classify these pixels in the overlapped masks.
- Removing the black strip in the images was also difficult. We were using various stitching algorithms to remove the black strip and stitch the images but in doing so we realized that some of the mask in the form of encoded pixels contained this black strip.

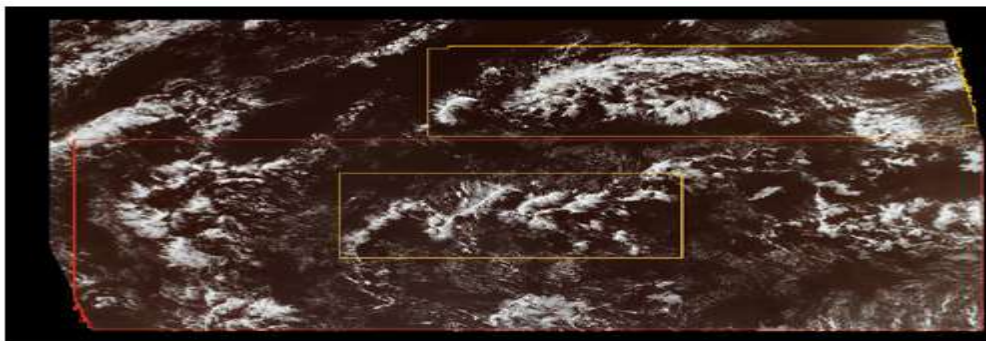


Figure 1: Flow chart for the Solution Approach

❖ METHODOLOGY

All the below steps shown in Figure-2 were implemented in Python Programming language using Keras Deep learning library, Matterport Mask-RCNN implementation[8] and Segmentation Models[7] on a Nvidia 1070 GTX GPU and Google Colab kernels.

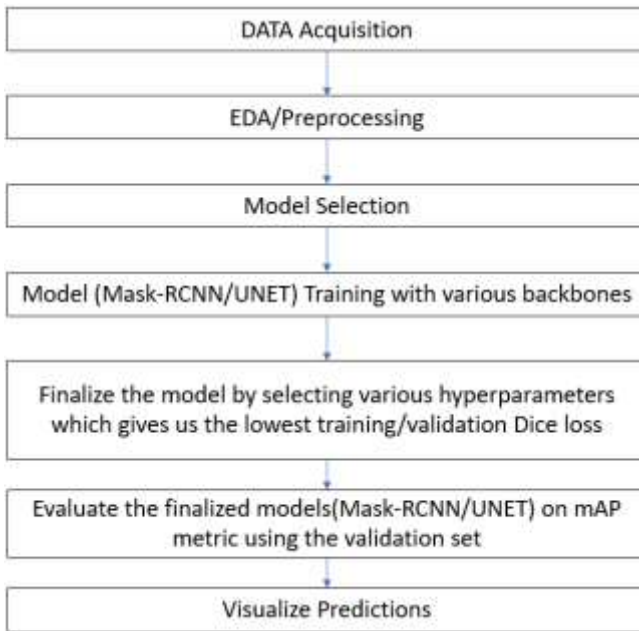


Figure 2: Flow chart for the Solution Approach

1. Data Acquisition

Data set has been taken from kaggle competition “Understanding Clouds from Satellite Images” which is acquired by NASA Worldwide. Data set can be found https://www.kaggle.com/c/understanding_cloud_organization/data.

2. EDA and Preprocessing

- The experimental data considered in this project consists of a ‘train_images’ folder which has the 5546 training images with size 2100*1400 and a train.csv file containing the run length encoded segmentations for each image-label pair in the ‘train_images’.
- Taking a closer look at the train.csv shown in figure-3 file we can see that:
 - For each image from the training dataset there are 4 lines for each type of clouds.
 - ‘Image_Label’ is a concatenation of the image filename and a cloud type.
 - If a certain type of clouds is present on the image, the ‘EncodedPixels’ column is non-null and contains the segmentation map for the corresponding cloud type.

Index	Image_Label	EncodedPixels
0	0011105.jpg_Fish	264918 937 266318 937 267718 937 269118 937 270518 937 271918 937 273318 937 274718 937 276118 937 277518 937 278918 937 280318 937 281718 937 283118 937 284518 937 285918 937 287318 937 288718 937 290118 937 291518 937 292918 937 294318 937 295718 937 297118 937 298518 937 299918 937 301318 937 302718 937 304118 937 305518 937 306918 937 308318 937 309718 937 311118 937 312518 937 313918 937 315318 937 316718 937 318118 937 319518 937 320918 937 322318 937 323718 937 325118 937 326518 937 327918 937 329318 937 330718 937 332118 937 333518 937 334918 937 336318 937 337718 937 339118 937 340518 937 341918 937 343318 937 344718 937 346118 937 347518 937 348918 937 350318 937 351718 937 353118 937 354518 937 355918 937 357318 937 358718 937 360118 937 361518 937 362918 937 364318 937 365718 937 367118 937 368518 937 369918 937 371318 937 372718 937 374118 937 375518 937 376918 937 378318 937 379718 937 381118 937 382518 937 383918 937 385318 937 386718 937 388118 937 389518 937 390918 937 392318 937 393718 937 395118 937 396518 937 397918 937 399318 937 400718 937 402118 937 403518 937 404918 937 406318 937 407718 937 409118 937 410518 937 411918 937 413318 937 414718 937 416118 937 417518 937 418918 937 420318 937 421718 937 423118 937 424518 937 425918 937 427318 937 428718 937 430118 937 431518 937 432918 937 434318 937 435718 937 437118 937 438518 937 439918 937 441318 937 442718 937 444118 937 445518 937 446918 937 448318 937 449718 937 451118 937 452518 937 453918 937 455318 937 456718 937 458118 937 459518 937 460918 937 462318 937 463718 937 465118 937 466518 937 467918 937 469318 937 470718 937 472118 937 473518 937 474918 937 476318 937 477718 937 479118 937 480518 937 481918 937 483318 937 484718 937 486118 937 487518 937 488918 937 490318 937 491718 937 493118 937 494518 937 495918 937 497318 937 498718 937 500118 937 501518 937 502918 937 504318 937 505718 937 507118 937 508518 937 509918 937 511318 937 512718 937 514118 937 515518 937 516918 937 518318 937 519718 937 521118 937 522518 937 523918 937 525318 937 526718 937 528118 937 529518 937 530918 937 532318 937 533718 937 535118 937 536518 937 537918 937 539318 937 540718 937 542118 937 543518 937 544918 937 546318 937 547718 937 549118 937 550518 937 551918 937 553318 937 554718 937 556118 937 557518 937 558918 937 560318 937 561718 937 563118 937 564518 937 565918 937 567318 937 568718 937 570118 937 571518 937 572918 937 574318 937 575718 937 577118 937 578518 937 579918 937 581318 937 582718 937 584118 937 585518 937 586918 937 588318 937 589718 937 591118 937 592518 937 593918 937 595318 937 596718 937 598118 937 599518 937 600918 937 602318 937 603718 937 605118 937 606518 937 607918 937 609318 937 610718 937 612118 937 613518 937 614918 937 616318 937 617718 937 619118 937 620518 937 621918 937 623318 937 624718 937 626118 937 627518 937 628918 937 630318 937 631718 937 633118 937 634518 937 635918 937 637318 937 638718 937 640118 937 641518 937 642918 937 644318 937 645718 937 647118 937 648518 937 649918 937 651318 937 652718 937 654118 937 655518 937 656918 937 658318 937 659718 937 661118 937 662518 937 663918 937 665318 937 666718 937 668118 937 669518 937 670918 937 672318 937 673718 937 675118 937 676518 937 677918 937 679318 937 680718 937 682118 937 683518 937 684918 937 686318 937 687718 937 689118 937 690518 937 691918 937 693318 937 694718 937 696118 937 697518 937 698918 937 700318 937 701718 937 703118 937 704518 937 705918 937 707318 937 708718 937 710118 937 711518 937 712918 937 714318 937 715718 937 717118 937 718518 937 719918 937 721318 937 722718 937 724118 937 725518 937 726918 937 728318 937 729718 937 731118 937 732518 937 733918 937 735318 937 736718 937 738118 937 739518 937 740918 937 742318 937 743718 937 745118 937 746518 937 747918 937 749318 937 750718 937 752118 937 753518 937 754918 937 756318 937 757718 937 759118 937 760518 937 761918 937 763318 937 764718 937 766118 937 767518 937 768918 937 770318 937 771718 937 773118 937 774518 937 775918 937 777318 937 778718 937 780118 937 781518 937 782918 937 784318 937 785718 937 787118 937 788518 937 789918 937 791318 937 792718 937 794118 937 795518 937 796918 937 798318 937 799718 937 801118 937 802518 937 803918 937 805318 937 806718 937 808118 937 809518 937 810918 937 812318 937 813718 937 815118 937 816518 937 817918 937 819318 937 820718 937 822118 937 823518 937 824918 937 826318 937 827718 937 829118 937 830518 937 831918 937 833318 937 834718 937 836118 937 837518 937 838918 937 840318 937 841718 937 843118 937 844518 937 845918 937 847318 937 848718 937 850118 937 851518 937 852918 937 854318 937 855718 937 857118 937 858518 937 859918 937 861318 937 862718 937 864118 937 865518 937 866918 937 868318 937 869718 937 871118 937 872518 937 873918 937 875318 937 876718 937 878118 937 879518 937 880918 937 882318 937 883718 937 885118 937 886518 937 887918 937 889318 937 890718 937 892118 937 893518 937 894918 937 896318 937 897718 937 899118 937 900518 937 901918 937 903318 937 904718 937 906118 937 907518 937 908918 937 910318 937 911718 937 913118 937 914518 937 915918 937 917318 937 918718 937 920118 937 921518 937 922918 937 924318 937 925718 937 927118 937 928518 937 929918 937 931318 937 932718 937 934118 937 935518 937 936918 937 938318 937 939718 937 941118 937 942518 937 943918 937 945318 937 946718 937 948118 937 949518 937 950918 937 952318 937 953718 937 955118 937 956518 937 957918 937 959318 937 960718 937 962118 937 963518 937 964918 937 966318 937 967718 937 969118 937 970518 937 971918 937 973318 937 974718 937 976118 937 977518 937 978918 937 980318 937 981718 937 983118 937 984518 937 985918 937 987318 937 988718 937 990118 937 991518 937 992918 937 994318 937 995718 937 997118 937 998518 937 1000000 937
1	0011165.jpg_Flower	1355585 1002 1358965 1002 1358365 1002 1359765 1002 1361165 1002 1362565 1002 1363965 1002 1365365 1002 1366765 1002 1368165 1002 1369565 1002 1370965 1002 1372365 1002 1373765 1002 1375165 1002 1376565 1002 1377965 1002 1379365 1002 1380765 1002 1382165 1002 1383565 1002 1384965 1002 1386365 1002 1387765 1002 1389165 1002 1390565 1002 1391965 1002 1393365 1002 1394765 1002 1396165 1002 1397565 1002 1398965 1002 1400365 1002 1401765 1002 1403165 1002 1404565 1002 1405965 1002 1407365 1002 1408765 1002 1410165 1002 1411565 1002 1412965 1002 1414365 1002 1415765 1002 1417165 1002 1418565 1002 1419965 1002 1421365 1002 1422765 1002 1424165 1002 1425565 1002 1426965 1002 1428365 1002 1429765 1002 1431165 1002 1432565 1002 1433965 1002 1435365 1002 1436765 1002 1438165 1002 1439565 1002 1440965 1002 1442365 1002 1443765 1002 1445165 1002 1446565 1002 1447965 1002 1449365 1002 1450765 1002 1452165 1002 1453565 1002 1454965 1002 1456365 1002 1457765 1002 1459165 1002 1460565 1002 1461965 1002 1463365 1002 1464765 1002 1466165 1002 1467565 1002 1468965 1002 1470365 1002 1471765 1002 1473165 1002 1474565 1002 1475965 1002 1477365 1002 1478765 1002 1480165 1002 1481565 1002 1482965 1002 1484365 1002 1485765 1002 1487165 1002 1488565 1002 1489965 1002 1491365 1002 1492765 1002 1494165 1002 1495565 1002 1496965 1002 1498365 1002 1499765 1002 1501165 1002 1502565 1002 1503965 1002 1505365 1002 1506765 1002 1508165 1002 1509565 1002 1510965 1002 1512365 1002 1513765 1002 1515165 1002 1516565 1002 1517965 1002 1519365 1002 1520765 1002 1522165 1002 1523565 1002 1524965 1002 1526365 1002 1527765 1002 1529165 1002 1530565 1002 1531965 1002 1533365 1002 1534765 1002 1536165 1002 1537565 1002 1538965 1002 1540365 1002 1541765 1002 1543165 1002 1544565 1002 1545965 1002 1547365 1002 1548765 1002 1550165 1002 1551565 1002 1552965 1002 1554365 1002 1555765 1002 1557165 1002 1558565 1002 1559965 1002 1561365 1002 1562765 1002 1564165 1002 1565565 1002 1566965 1002 1568365 1002 1569765 1002 1571165 1002 1572565 1002 1573965 1002 1575365 1002 1576765 1002 1578165 1002 1579565 1002 1580965 1002 1582365 1002 1583765 1002 1585165 1002 1586565 1002 1587965 1002 1589365 1002 1590765 1002 1592165 1002 1593565 1002 1594965 1002 1596365 1002 1597765 1002 1599165 1002 1600565 1002 1601965 1002 1603365 1002 1604765 1002 1606165 1002 1607565 1002 1608965 1002 1610365 1002 1611765 1002 1613165 1002 1614565 1002 1615965 1002 1617365 1002 1618765 1002 1620165 1002 1621565 1002 1622965 1002 1624365 1002 1625765 1002 1627165 1002 1628565 1002 1629965 1002 1631365 1002 1632765 1002 1634165 1002 1635565 1002 1636965 1002 1638365 1002 1639765 1002 1641165 1002 1642565 1002 1643965 1002 1645365 1002 1646765 1002 1648165 1002 1649565 1002 1650965 1002 1652365 1002 1653765 1002 1655165 1002 1656565 1002 1657965 1002 1659365 1002 1660765 1002 1662165 1002 1663565 1002 1664965 1002 1666365 1002 1667765 1002 1669165 1002 1670565 1002 1671965 1002 1673365 1002 1674765 1002 1676165 1002 1677565 1002 1678965 1002 1680365 1002 1681765 1002 1683165 1002 1684565 1002 1685965 1002 1687365 1002 1688765 1002 1690165 1002 1691565 1002 1692965 1002 1694365 1002 1695765 1002 1697165 1002 1698565 1002 1699965 1002 1701365 1002 1702765 1002 1704165 1002 1705565 1002 1706965 1002 1708365 1002 1709765 1002 1711165 1002 1712565 1002 1713965 1002 1715365 1002 1716765 1002 1718165 1002 1719565 1002 1720965 1002 1722365 1002 1723765 1002 1725165 1002 1726565 1002 1727965 1002 1729365 1002 1730765 1002 1732165 1002 1733565 1002 1734965 1002 1736365 1002 1737765 1002 1739165 1002 1740565 1002 1741965 1002 1743365 1002 1744765 1002 1746165 1002 1747565 1002 1748965 1002 1750365 1002 1751765 1002 1753165 1002 1754565 1002 1755965 1002 1757365 1002 1758765 1002 1760165 1002 1761565 1002 1762965 1002 1764365 1002 1765765 1002 1767165 1002 1768565 1002 1769965 1002 1771365 1002 1772765 1002 1774165 1002 1775565 1002 1776965 1002 1778365 1002 1779765 1002 1781165 1002 1782565 1002 1783965 1002 1785365 1002 1786765 1002 1788165 1002 1789565 1002 1790965 1002 1792365 1002 1793765 1002 1795165 1002 1796565 1002 1797965 1002 1799365 1002 1800765 1002 1802165 1002 1803565 1002 1804965 1002 1806365 1002 1807765 1002 1809165 1002 1810565 1002 1811965 1002 1813365 1002 1814765 1002 1816165 1002 1817565 1002 1818965 1002 1820365 1002 1821765 1002 1823165 1002 1824565 1002 1825965 1002 1827365 1002 1828765 1002 1830165 1002 1831565 1002 1832965 1002 1834365 1002 1835765 1002 1837165 1002 1838565 1002 1839965 1002 1841365 1002 1842765 1002 1844165 1002 1845565 1002 1846965 1002 1848365 1002 1849765 1002 1851165 1002 1852565 1002 1853965 1002 1855365 1002 1856765 1002 1858165 1002 1859565 1002 1860965 1002 1862365 1002 1863765 1002 1865165 1002 1866565 1002 1867965 1002 1869365 1002 1870765 1002 1872165 1002 1873565 1002 1874965 1002 1876365 1002 1877765 1002 1879165 1002 1880565 1002 1881965 1002 1883365 1002 1884765 1002 1886165 1002 1887565 1002 1888965 1002 1890365 1002 1891765 1002 1893165 1002 1894565 1002 1895965 1002 1897365 1002 1898765 1002 1900165 1002 1901565 1002 1902965 1002 1904365 1002 1905765 1002 1907165 1002 1908565 1002 1909965 1002 1911365 1002 1912765 1002 1914165 1002 1915565 1002 1916965 1002 1918365 1002 1919765 1002 1921165 1002 1922565 1002 1923965 1002 1925365 1002 1926765 1002 1928165 1002 1929565 1002 1930965 1002 1932365 1002 1933765 1002 1935165 1002 1936565 1002 1937965 1002 1939365 1002 1940765 1002 1942165 1002 1943565 1002 1944965 1002 1946365 1002 1947765 1002 1949165 1002 1950565 1002 1951965 1002 1953365 1002 1954765 1002 1956165 1002 1957565 1002 1958965 1002 1960365 1002 1961765 1002 1963165 1002 1964565 1002 196

Figure 3: contents of train.csv file.

- We have total 22184 Image labels (segmentation masks) and while looking for null values shown in figure-4, we observed that 10348 rows don't have encoded pixels i.e. they have empty segmentation maps, as a result we have deleted those data entries and will be working with 11836 image labels.

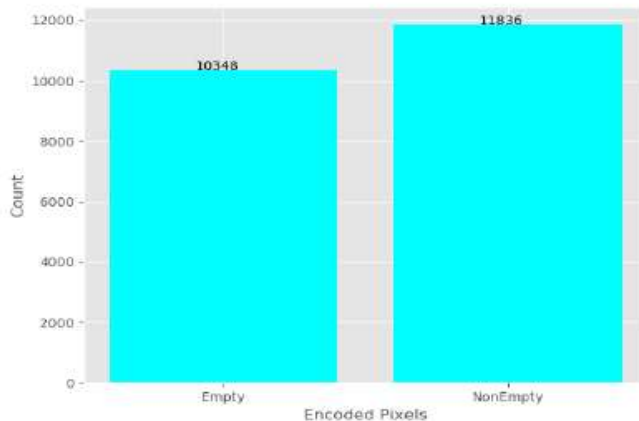


Figure 4: Count plot showing empty and non-empty segmentation masks.

- Counting the number of labels of each cloud type (Figure-5) we found that there are 2781 of Fish, 2365 of Flower, 2939 of Gravel and 3751 of Sugar observations respectively. From the plots below we can see that the dataset is somewhat balanced making the task a bit easier.

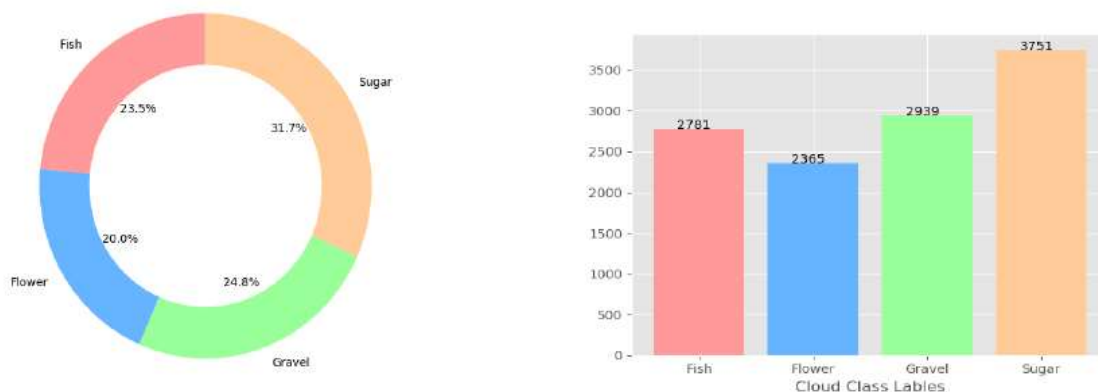


Figure 5: Donut plot(left) and count plot(right) showing the Cloud class distribution in the dataset.

- We wanted to see the number of mask labels per image and observed that 2372 Images which has two classes, 1560 Images with three classes, 266 images with all the four classes and 1348 with only one mask. We can conclude from figure-6 that most of the times we have 2 or 3 types of cloud formations in one image, all the 4 types of cloud formation in one image is very rare. Only one type of cloud formation in the image is also somewhat common

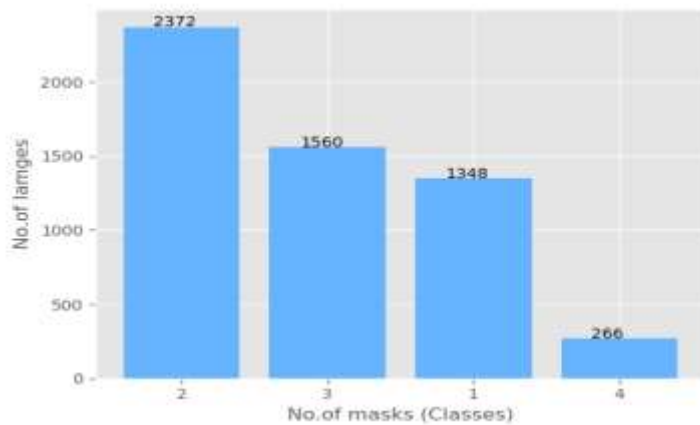


Figure 6: Bar plot showing frequency of labels per image.

- We also wanted to see which combination of cloud formations which occur frequently and combinations that hardly appears together. To do this we used a simple datamining algorithm called Frequent Patter Mining and plotted the results shown in Figure-7. We can state from the figure below that Sugar cloud formation frequently appears together in the images with Gravel or Fish cloud formation. Sugar also appears with Flower cloud formation but is less frequent. Gravels and Fish cloud formation also appears with other cloud formation. Sugar, Gravel, and Fish also appears all together in some instances. Flower tends to occur less frequently with other clouds, and the combination of Gravel and Flower occurs but at much less frequency compared to others. In fact, Sugar, Gravel, and Fish appear all together more frequently than Grave and Flower. However, it's not like Flower cloud formation never occurs with other cloud formation, just occurs less frequently compared to others. In summary, they are all combination of cloud formations appearing together is a possibility, and the combinations between Sugar, Fish, and Gravel are more likely than with Flower cloud formation.

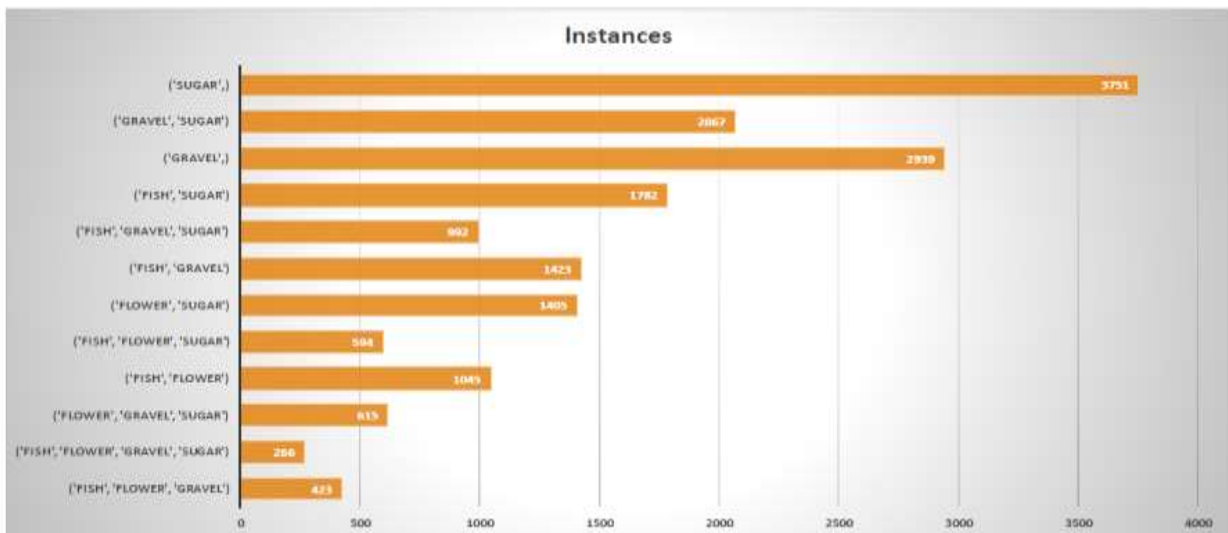


Figure 7: Plot showing frequent patterns of the cloud formations.

- In preprocessing after the removal of missing values stated above, we resized the images into sizes 512*512 and 320*320 and created custom image data generator classes for the two models(Mask-RCNN and UNET) respectively. We also used a train-validation split with a ratio of 90:10 for both the models.

3. Model Selection

Reading various literature, upon further research and with the help of the professor we decided to implement two artificial deep neural networks, Mask-RCNN and UNET.

3.1 MASK RCNN

Mask RCNN is an artificial deep neural network aimed to solve object instance segmentation problem in machine learning or computer vision. Mask R-CNN efficiently detects objects in an image while simultaneously generating a high-quality segmentation mask for each instance. It extends Faster R-CNN by adding a branch for predicting a binary object mask in parallel with the existing branch for bounding box recognition [1].

There are two stages of Mask R-CNN shown in figure-8. The first stage, called an RPN, proposes potential regions where there might be objects for a given image. The second stage, extracts features using ROI Pool from each candidate box(proposed potential regions), performs classification along with bounding box regression and generates a binary mask for each ROI. Both stages are connected to an FPN style deep neural network usually termed as the backbone structure. The backbone is usually a pretrained network like ConvNet, VGG or ResNet.

In the first stage the RPN scans all the feature maps and proposes regions which may contain objects. To bind features to its raw image location efficiently the network uses a set of boxes with predefined locations and scales relative to the image called Anchors. The true masks and the bounding boxes are assigned to individual anchors according to some preset IoU value. The RPN uses anchors with different scales bind to different levels of feature map to figure out where the feature map 'should' get an object and what size of its bounding box is [3].

In the second stage the regions proposed by the first stage are assigned to several specific areas of a feature map, scans these areas, and generates object classes, bounding boxes and masks. This procedure looks similar to RPN, but the difference is that, stage-two uses ROIAlign to locate the relevant areas of feature map instead of anchors, and there is an additional branch which generates masks for each object in pixel level (pixel level classification).

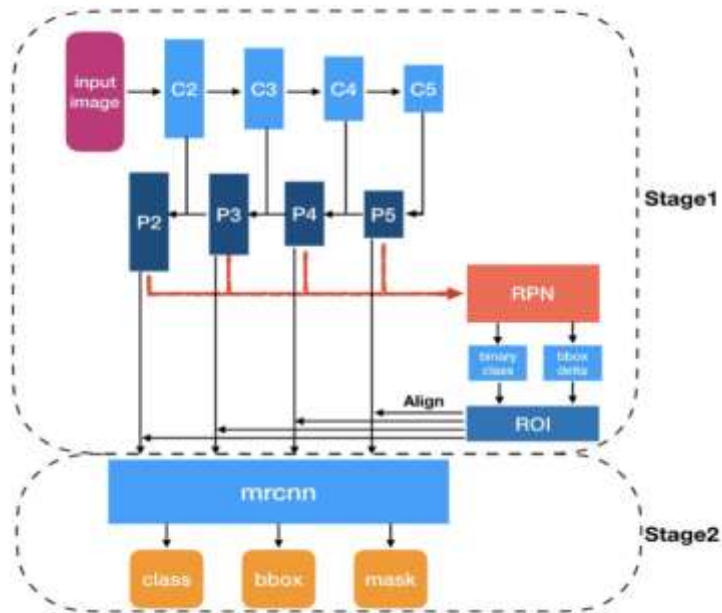


Figure 8: Mask – RCNN architecture

3.2 U-NET

The Artificial deep neural network takes the shape of an ‘U’ hence termed as U-net. The U-Net architecture for the most part is symmetric and consists of two major parts shown in Figure-9, the left/down part which is a contracting/downsampling path and the right/up part which is an expanding/upsampling path.

The contracting path is similar to an encoder and consists of several convolution layers followed by an activation function(ReLU), batch norm and max-pooling layers. Its purpose is to capture the context of the input image via a compact feature map in order to perform segmentation. This coarse contextual information will then be transferred to the upsampling path by means of skip connections [5]. The encoder part is referred as a backbone of the U-Net and is usually a pretrained network like VGG, ResNet, InceptionNet, EfficientNet, DenseNet, etc.

The expanding path is similar to a decoder which consists of deconvolution layers (up-sampling) and concatenation followed by a symmetrical number of convolution layers with an activation function (ReLU) and batch normalization to that of the encoder part. The decoder part’s purpose is to enable precise localization combined with contextual information from the contracting path. This step is done to retain boundary information (spatial information) despite down sampling and max-pooling performed in the encoder stage [2].

In general terms the encoder part encodes the input image into feature representation at multiple different levels and the decoder part semantically projects the discriminative lower resolution features learnt by the encoder into higher resolution pixel space to get a dense classification.

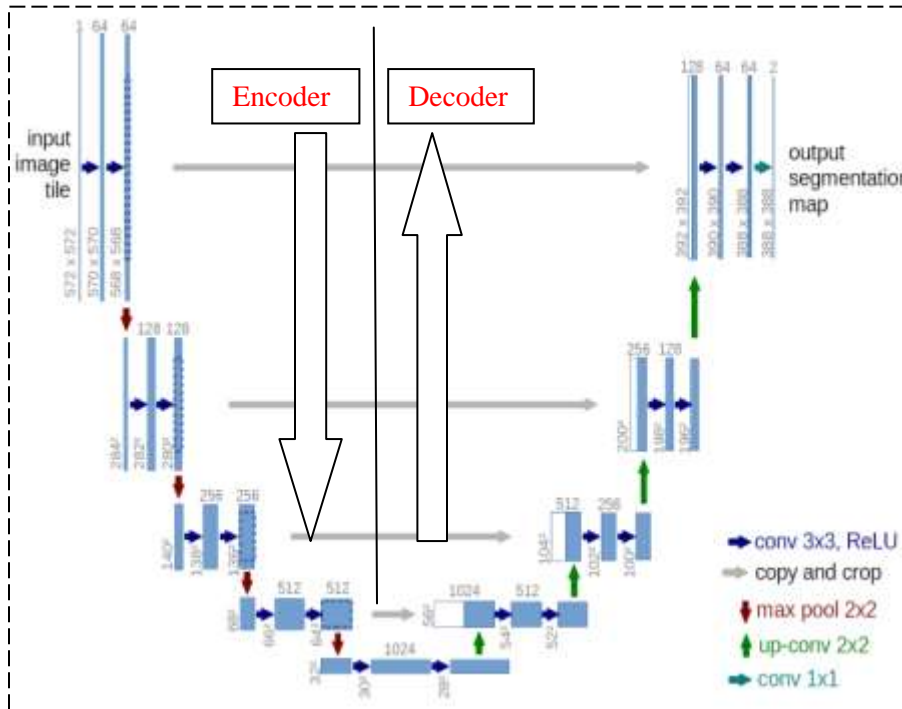


Figure 9: U-Net architecture

4. Model Training

- The Mask-RCNN model designed used a ResNet-101 architecture as a backbone encoder and was initialized with COCO (Common Objects with Context) weights. Most of the hyperparameters used default values except some parameters like RPN Anchor Scales, Steps per epoch, etc, were tuned accordingly. We split the dataset into 90% train and 10% validation sets and images were resized to 512*512 to speed up the computation. Image augmentations were used during training due to the small dataset. The entire model was train and validated using Adam optimizer with average binary cross-entropy loss and Dice loss(1-Dice Coefficient) as the performance metric for 25 epochs on a learning rate of 0.0002. Initially for the first epoch only the heads (top layer) were trained and for remaining epochs all the layers were trained. Training and validation loss plots for the model are shown in figure-10. The top plot entails the combined loss for all the different sub networks(RPN, class, bounding box and mask) in the Mask-RCNN model. The bottom plot shows the Dice loss for the generated masks.

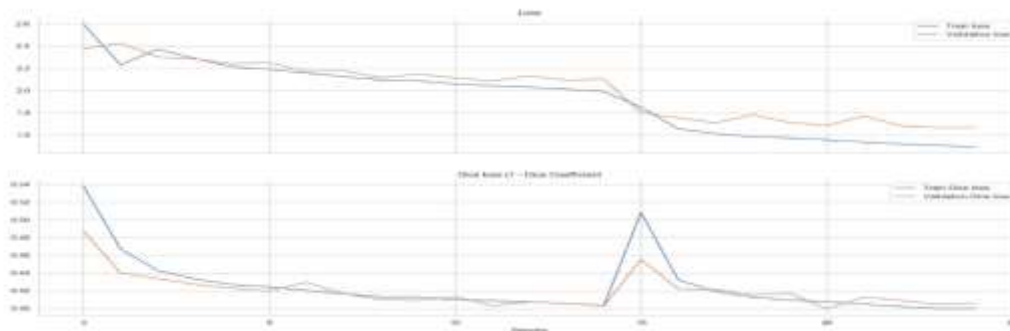


Figure 10: Loss plots for Mask-RCNN

- The implemented UNET model used an EfficientNet-b3 architecture as a backbone encoder and was initialized with Imagenet weights. We split the dataset into 90% train and 10% validation sets and images were resized to 320*320 to reduce the training time. We relied heavily on image augmentations to ensure the model could generalize well even though the images were reduced to a very small size. We used a modified Adam optimizer so that several hyperparameters and learning rate can be improved and optimized adaptively. The entire model was train and validated using the modified Adam optimizer with balanced cross-entropy dice loss(BCE) and Dice Coefficient as the performance metric for 15 epochs with an initial learning rate of 0.002. Unlike Mask-RCNN all the layers were trained for each epoch. Training and validation loss plots for the UNET model are shown in figure-11.

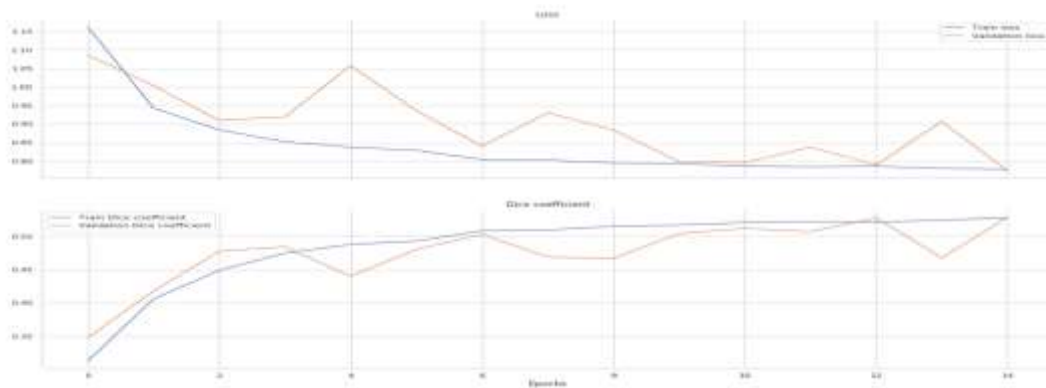


Figure 11: Loss plots for UNET

5. Model Evaluation

Both the models were evaluated on MAP (Mean Average Precision) metric using the validation set and the results are show in Table-1

➤ Mean Average Precision:

Average precision is a popular metric in measuring the accuracy of object detectors like Masked RCNN. Average precision is a measure that combines recall and precision for ranked retrieval results and the Mean Average Precision (MAP) is the arithmetic mean of the average precision values for an information retrieval system over a set of Q query topics.

$$MAP = \frac{\sum_{q=1}^Q AveP(q)}{Q}$$

- MAP for two models:

Mask-RCNN	U-Net
0.33	0.31

6. PREDECTED RESULTS

- **Mask RCNN:** Model is initialized with its best weights and masks along with the cloud class were predicted for some sample images in the validation set.

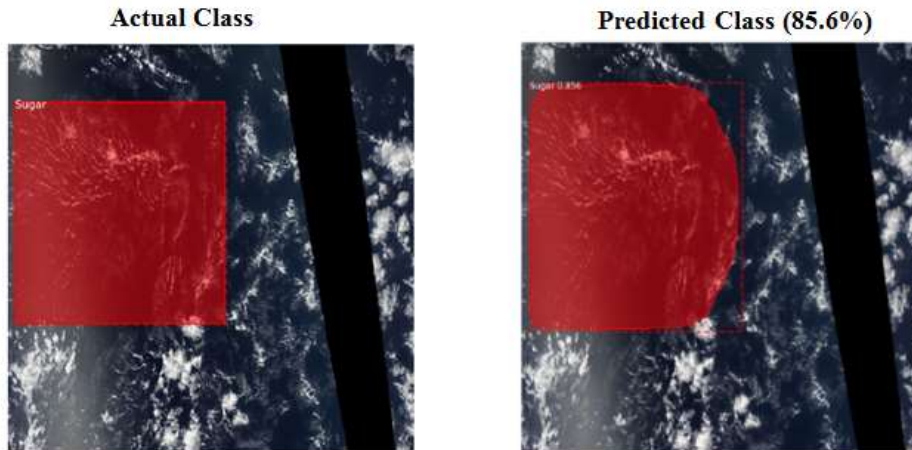


Figure 12: Prediction for one class label

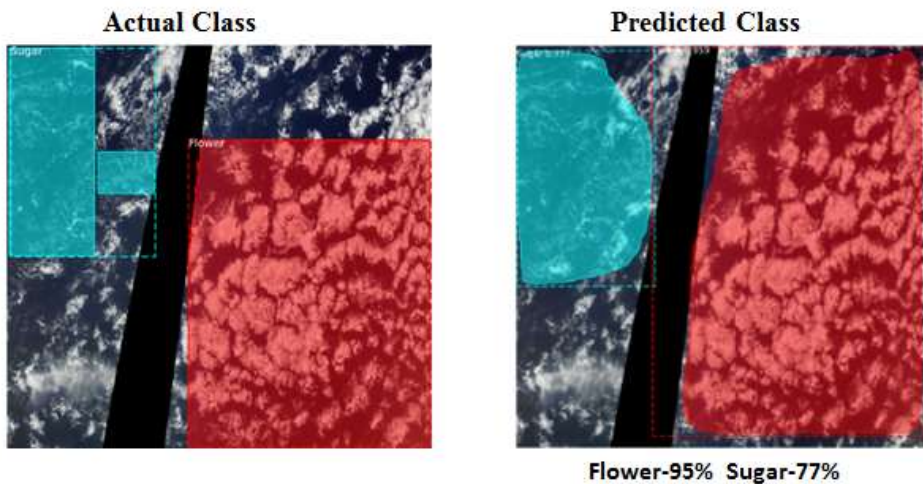


Figure 13: Prediction for Two class label

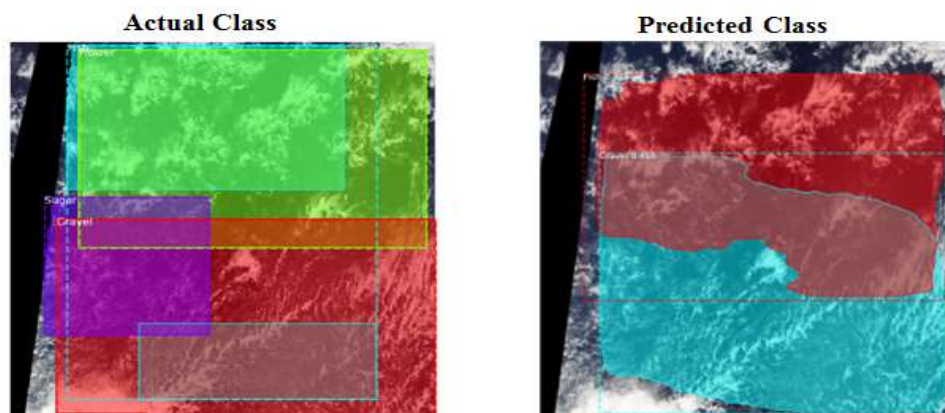


Figure 14: Prediction for Three class label

- **U-Net:** Model is initialized with its best weights and masks along with the cloud class were predicted for some sample images in the validation set after post processing.

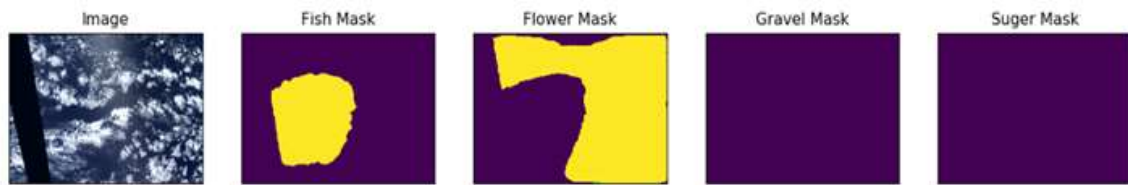


Figure 15: Prediction for one class label



Figure 16: Prediction for Two class labels



Figure 17: Prediction for Three class label

7. CONCLUSION

Climate change in the present day is an important issue and shallow clouds play a huge role in determining the earth's climate [6]. Classifying different cloud types can help researchers build better climate models. In this project we built two artificial deep neural network models Mask-RCNN and UNET to classify 4 cloud organization patterns namely Fish, Flower, Sugar and Gravel from satellite images. We used the concept of transfer learning to train both models. The Mask-RCNN was trained using a ResNet-101 backbone with Adam optimizer and the UNET model was trained using EfficientNet-b3 backbone with a modified Adam optimizer. Both the models were trained using Dice loss as the performance metric and are evaluated on MAP (mean average Precision) metric. It was observed that the Mask-RCNN performed slightly better with a MAP of 0.33 on the validation set compared to a MAP score 0.31 of the UNET model. Although we used the same data this is not a fair comparison as many parameters were different input images sizes, backbones and main being the number of epochs trained (25 for Mask-RCNN and 15 for UNET). We also observed that it was difficult for both the models to predict mask which were overlapping each other, and which contained a lot of background noise, that being said with more training time and proper image preprocessing both the models can perform much better.

8. FUTURE WORK

The avenues for future work include the following:

- Improving the predictive results by using ensembles of models.
- Use Cross validation to better assess the effectiveness of the model.
- A better statistical approach in choosing the right thresholds in post processing.
- Better Hyper parameter Tuning and using different loss functions.
- Use different model architectures and Backbones(pre-trained networks),
- Use the classifier with explain ability techniques like Gradient-weighted Class Activation Mapping to generate a baseline.

9. REFERENCES

- [1] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).
- [2] Ronneberger, O., Fischer, P., & Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Springer, Cham.
- [3] https://wiki.ubc.ca/CNNs_in_Image_Segmentation
- [4] Rasp, S., Schulz, H., Bony, S., & Stevens, B. (2019). Combining crowd-sourcing and deep learning to understand meso-scale organization of shallow convection. *arXiv preprint arXiv:1906.01906*.
- [5] Zantedeschi, V., Falasca, F., Douglas, A., Strange, R., Kusner, M. J., & Watson-Parris, D.

- (2019). Cumulo: A Dataset for Learning Cloud Classes. *arXiv preprint arXiv:1911.04227*.
- [6] https://www.kaggle.com/c/understanding_cloud_organization/data# =
- [7] Segmentation models: https://github.com/qubvel/segmentation_models
- [8] Mask RCNN: https://github.com/matterport/Mask_RCNN
- [9] <https://engineering.matterport.com/splash-of-color-instance-segmentation-with-mask-r-cnn-and-tensorflow-7c761e238b46>