Concept correlation between C2 (concept bottleneck model) and hybrid network () This code averages each validation set e.g all 12 instances of rock validation numbers 1,2,3

In [ ]:
```
import os
import pandas as pd
import fnmatch
import pandoc

root = "C:/Users/c21012241/Dropbox"

### 13 Features
path =  root + "/13 Features - Binary Crystals/\
C2 LR 10^-3 E 200 MB 1024 - H LR 10^-3 E 50 MiniBatch 1024 - 13U LR 10^-3 E 200 MB 1024 - 12 of 12"

#path =  root + "13 Features - Continuous Crystals/\
#C2 LR 10^-3 Epochs 200 MiniBatch 1024 - Hybrid LR 10^-3 Epochs 50 MiniBatch 1024 - 12 of 12"

#path =  root + "/13 Features - Binary Crystals/\
#C2 LR10^-3 E200 MB1025 - H LR10^-3 E15 MB1024 - 13U LR10^-3 E200 MB1024 - 12 of 12"

### 12 Features
#path =  root +"/12 Features - Binary Crystals + No Brightness/\
#C2 LR 10^-3 E 200 MB 1024 - H LR 10^-3 E 50 MiniBatch 1024 - 12U LR 10^-3 E 200 MB 1024 - 12 of 12"

#path =  root +"/12 Features - Continuous Crystals + No Bright/\
#C2 LR10^-3 E200 MB1024 - H LR10^-3 E50 MB1024 - 12U LR10^-3 E200 MB1024 - 12of12"

### Trained with re-rated features dataset
#path =  root +"/Re-rated expertFeatures - 13 - Binary/\
#C2 LR 10^-3 E 200 MB 1024 - H LR 10^-3 E 50 MiniBatch 1024 - 13U LR 10^-3 E 200 MB 1024 - 12 of 12"
```

In [ ]:
```
C2_Predicted_Features = []
hybrid_13_Nodes = []
Val_C2_1_2_3 = []
Val_C2_4_5_6 = []
Val_C2_7_8_9 = []
Val_C2_10_11_12 = []
Val_C2_1_4_7 = []
Val_C2_5_8_10 = []
Val_C2_2_9_11 = []
Val_C2_3_6_12 = []
Val_C2_1_6_9 = []
Val_C2_2_7_10 = []
Val_C2_3_8_11 = []
Val_C2_4_5_12 = []
Val_Hybrid_1_2_3 = []
Val_Hybrid_4_5_6 = []
Val_Hybrid_7_8_9 = []
Val_Hybrid_10_11_12 = []
Val_Hybrid_1_4_7 = []
Val_Hybrid_5_8_10 = []
Val_Hybrid_2_9_11 = []
Val_Hybrid_3_6_12 = []
Val_Hybrid_1_6_9 = []
Val_Hybrid_2_7_10 = []
Val_Hybrid_3_8_11 = []
Val_Hybrid_4_5_12 = []
```

In [ ]:
```
keyword_C2 = "*C2 Network - predicted_features*"
keyword_Hybrid = "*netHybrid - Average 13 Node Activations of 13x256 matrix*"

#keyword_Hybrid = "*netHybrid-Av13NodeActs*"
#keyword_C2 = "*C2-PredFeatures*"
```

In [ ]:
```
keyword_01_02_03 = "*Val_1 2 3*"
keyword_04_05_06 = "*Val_4 5 6*"
keyword_07_08_09 = "*Val_7 8 9*"
keyword_10_11_12 = "*Val_10 11 12*"
keyword_01_04_07 = "*Val_1 4 7*"
keyword_05_08_10 = "*Val_5 8 10*"
keyword_02_09_11 = "*Val_2 9 11*"
keyword_03_06_12 = "*Val_3 6 12*"
keyword_01_06_09 = "*Val_1 6 9*"
keyword_02_07_10 = "*Val_2 7 10*"
keyword_03_08_11 = "*Val_3 8 11*"
keyword_04_05_12 = "*Val_4 5 12*"
```

In [ ]:
```
# Walk through the root folder into sub folders
for root, dirs, files in os.walk(path):
# If a file name matches the C2 keyword, add it to the list
    for filename in fnmatch.filter(files, keyword_C2):
        file_path = os.path.join(root, filename)
        C2_Predicted_Features.append(file_path)

# Walk through the root folder into sub folders
for root, dirs, files in os.walk(path):
# If a file name matches hybrid network keyword, add it to the list
    for filename in fnmatch.filter(files, keyword_Hybrid):
        file_path = os.path.join(root, filename)
        hybrid_13_Nodes.append(file_path)

# Sort the list based on the time stamp
C2_Predicted_Features.sort(key=os.path.getmtime)
hybrid_13_Nodes.sort(key=os.path.getmtime)
```

In [ ]:

```python
# Walk through the sorted list and if a keyword matches then add it to the relevant list
for file in C2_Predicted_Features:
    if fnmatch.fnmatch(file, keyword_01_02_03):
        df = pd.read_csv(file, header=None)
        Val_C2_1_2_3.append(df)
    elif fnmatch.fnmatch(file, keyword_04_05_06):
        df = pd.read_csv(file, header=None)
        Val_C2_4_5_6.append(df)
    elif fnmatch.fnmatch(file, keyword_07_08_09):
        df = pd.read_csv(file, header=None)
        Val_C2_7_8_9.append(df)
    elif fnmatch.fnmatch(file, keyword_10_11_12):
        df = pd.read_csv(file, header=None)
        Val_C2_10_11_12.append(df)
    elif fnmatch.fnmatch(file, keyword_01_04_07):
        df = pd.read_csv(file, header=None)
        Val_C2_1_4_7.append(df)
    elif fnmatch.fnmatch(file, keyword_05_08_10):
        df = pd.read_csv(file, header=None)
        Val_C2_5_8_10.append(df)
    elif fnmatch.fnmatch(file, keyword_02_09_11):
        df = pd.read_csv(file, header=None)
        Val_C2_2_9_11.append(df)
    elif fnmatch.fnmatch(file, keyword_03_06_12):
        df = pd.read_csv(file, header=None)
        Val_C2_3_6_12.append(df)
    elif fnmatch.fnmatch(file, keyword_01_06_09):
        df = pd.read_csv(file, header=None)
        Val_C2_1_6_9.append(df)
    elif fnmatch.fnmatch(file, keyword_02_07_10):
        df = pd.read_csv(file, header=None)
        Val_C2_2_7_10.append(df)
    elif fnmatch.fnmatch(file, keyword_03_08_11):
        df = pd.read_csv(file, header=None)
        Val_C2_3_8_11.append(df)
    elif fnmatch.fnmatch(file, keyword_04_05_12):
        df = pd.read_csv(file, header=None)
        Val_C2_4_5_12.append(df)
```

In [ ]:

```python
for file in hybrid_13_Nodes:
    if fnmatch.fnmatch(file, keyword_01_02_03):
        df = pd.read_csv(file, header=None)
        Val_Hybrid_1_2_3.append(df)
    elif fnmatch.fnmatch(file, keyword_04_05_06):
        df = pd.read_csv(file, header=None)
        Val_Hybrid_4_5_6.append(df)
    elif fnmatch.fnmatch(file, keyword_07_08_09):
        df = pd.read_csv(file, header=None)
        Val_Hybrid_7_8_9.append(df)
    elif fnmatch.fnmatch(file, keyword_10_11_12):
        df = pd.read_csv(file, header=None)
        Val_Hybrid_10_11_12.append(df)
    elif fnmatch.fnmatch(file, keyword_01_04_07):
        df = pd.read_csv(file, header=None)
        Val_Hybrid_1_4_7.append(df)
    elif fnmatch.fnmatch(file, keyword_05_08_10):
        df = pd.read_csv(file, header=None)
        Val_Hybrid_5_8_10.append(df)
    elif fnmatch.fnmatch(file, keyword_02_09_11):
        df = pd.read_csv(file, header=None)
        Val_Hybrid_2_9_11.append(df)
    elif fnmatch.fnmatch(file, keyword_03_06_12):
        df = pd.read_csv(file, header=None)
        Val_Hybrid_3_6_12.append(df)
    elif fnmatch.fnmatch(file, keyword_01_06_09):
        df = pd.read_csv(file, header=None)
        Val_Hybrid_1_6_9.append(df)
    elif fnmatch.fnmatch(file, keyword_02_07_10):
        df = pd.read_csv(file, header=None)
        Val_Hybrid_2_7_10.append(df)
    elif fnmatch.fnmatch(file, keyword_03_08_11):
        df = pd.read_csv(file, header=None)
        Val_Hybrid_3_8_11.append(df)
    elif fnmatch.fnmatch(file, keyword_04_05_12):
        df = pd.read_csv(file, header=None)
        Val_Hybrid_4_5_12.append(df)
```

In [ ]:

```python
# C2
# Sum and average each validation set, then save to a csv file
Av_C2_Val_1_2_3 = sum(Val_C2_1_2_3)/12
Av_C2_Val_1_2_3.to_csv(path + " " + keyword_C2.replace("*","") + " " + keyword_01_02_03.replace("*","") +'.csv',
                       header=None, index = False, encoding='utf-8')
Av_C2_Val_4_5_6 = sum(Val_C2_4_5_6)/12
Av_C2_Val_4_5_6.to_csv(path + " " + keyword_C2.replace("*","") + " " + keyword_04_05_06.replace("*","") +'.csv',
                       header=None, index = False, encoding='utf-8')
Av_C2_Val_7_8_9 = sum(Val_C2_7_8_9)/12
Av_C2_Val_7_8_9.to_csv(path + " " + keyword_C2.replace("*","") + " " + keyword_07_08_09.replace("*","") +'.csv',
                       header=None, index = False, encoding='utf-8')
Av_C2_Val_10_11_12 = sum(Val_C2_10_11_12)/12
Av_C2_Val_10_11_12.to_csv(path + " " + keyword_C2.replace("*","") + " " + keyword_10_11_12.replace("*","") +'.csv',
                          header=None, index = False, encoding='utf-8')
Av_C2_Val_1_4_7 = sum(Val_C2_1_4_7)/12
Av_C2_Val_1_4_7.to_csv(path + " " + keyword_C2.replace("*","") + " " + keyword_01_04_07.replace("*","") +'.csv',
                       header=None, index = False, encoding='utf-8')
Av_C2_Val_5_8_10 = sum(Val_C2_5_8_10)/12
Av_C2_Val_5_8_10.to_csv(path + " " + keyword_C2.replace("*","") + " " + keyword_05_08_10.replace("*","") +'.csv',
                        header=None, index = False, encoding='utf-8')
Av_C2_Val_2_9_11 = sum(Val_C2_2_9_11)/12
Av_C2_Val_2_9_11.to_csv(path + " " + keyword_C2.replace("*","") + " " + keyword_02_09_11.replace("*","") +'.csv',
                        header=None, index = False, encoding='utf-8')
Av_C2_Val_3_6_12 = sum(Val_C2_3_6_12)/12
Av_C2_Val_3_6_12.to_csv(path + " " + keyword_C2.replace("*","") + " " + keyword_03_06_12.replace("*","") +'.csv',
                        header=None, index = False, encoding='utf-8')
Av_C2_Val_1_6_9 = sum(Val_C2_1_6_9)/12
Av_C2_Val_1_6_9.to_csv(path + " " + keyword_C2.replace("*","") + " " + keyword_01_06_09.replace("*","") +'.csv',
                       header=None, index = False, encoding='utf-8')
Av_C2_Val_2_7_10 = sum(Val_C2_2_7_10)/12
Av_C2_Val_2_7_10.to_csv(path + " " + keyword_C2.replace("*","") + " " + keyword_02_07_10.replace("*","") +'.csv',
                        header=None, index = False, encoding='utf-8')
Av_C2_Val_3_8_11 = sum(Val_C2_3_8_11)/12
Av_C2_Val_3_8_11.to_csv(path + " " + keyword_C2.replace("*","") + " " + keyword_03_08_11.replace("*","") +'.csv',
                        header=None, index = False, encoding='utf-8')
Av_C2_Val_4_5_12 = sum(Val_C2_4_5_12)/12
Av_C2_Val_4_5_12.to_csv(path + " " + keyword_C2.replace("*","") + " " + keyword_04_05_12.replace("*","") +'.csv',
                        header=None, index = False, encoding='utf-8')
```

In [ ]:

```python
# Hybrid
# Sum and average each validation set, then save to a csv file
Av_Hybrid_Val_1_2_3 = sum(Val_Hybrid_1_2_3)/12
Av_Hybrid_Val_1_2_3.to_csv(path + " " + keyword_Hybrid.replace("*","") + " " + keyword_01_02_03.replace("*","") +'.csv',
                           header=None, index = False, encoding='utf-8')
Av_Hybrid_Val_4_5_6 = sum(Val_Hybrid_4_5_6)/12
Av_Hybrid_Val_4_5_6.to_csv(path + " " + keyword_Hybrid.replace("*","") + " " + keyword_04_05_06.replace("*","") +'.csv',
                           header=None, index = False, encoding='utf-8')
Av_Hybrid_Val_7_8_9 = sum(Val_Hybrid_7_8_9)/12
Av_Hybrid_Val_7_8_9.to_csv(path + " " + keyword_Hybrid.replace("*","") + " " + keyword_07_08_09.replace("*","") +'.csv',
                           header=None, index = False, encoding='utf-8')
Av_Hybrid_Val_10_11_12 = sum(Val_Hybrid_10_11_12)/12
Av_Hybrid_Val_10_11_12.to_csv(path + " " + keyword_Hybrid.replace("*","") + " " + keyword_10_11_12.replace("*","") +'.csv',
                              header=None, index = False, encoding='utf-8')
Av_Hybrid_Val_1_4_7 = sum(Val_Hybrid_1_4_7)/12
Av_Hybrid_Val_1_4_7.to_csv(path + " " + keyword_Hybrid.replace("*","") + " " + keyword_01_04_07.replace("*","") +'.csv',
                           header=None, index = False, encoding='utf-8')
Av_Hybrid_Val_5_8_10 = sum(Val_Hybrid_5_8_10)/12
Av_Hybrid_Val_5_8_10.to_csv(path + " " + keyword_Hybrid.replace("*","") + " " + keyword_05_08_10.replace("*","") +'.csv',
                            header=None, index = False, encoding='utf-8')
Av_Hybrid_Val_2_9_11 = sum(Val_Hybrid_2_9_11)/12
Av_Hybrid_Val_2_9_11.to_csv(path + " " + keyword_Hybrid.replace("*","") + " " + keyword_02_09_11.replace("*","") +'.csv',
                            header=None, index = False, encoding='utf-8')
Av_Hybrid_Val_3_6_12 = sum(Val_Hybrid_3_6_12)/12
Av_Hybrid_Val_3_6_12.to_csv(path + " " + keyword_Hybrid.replace("*","") + " " + keyword_03_06_12.replace("*","") +'.csv',
                            header=None, index = False, encoding='utf-8')
Av_Hybrid_Val_1_6_9 = sum(Val_Hybrid_1_6_9)/12
Av_Hybrid_Val_1_6_9.to_csv(path + " " + keyword_Hybrid.replace("*","") + " " + keyword_01_06_09.replace("*","") +'.csv',
                           header=None, index = False, encoding='utf-8')
Av_Hybrid_Val_2_7_10 = sum(Val_Hybrid_2_7_10)/12
Av_Hybrid_Val_2_7_10.to_csv(path + " " + keyword_Hybrid.replace("*","") + " " + keyword_02_07_10.replace("*","") +'.csv',
                            header=None, index = False, encoding='utf-8')
Av_Hybrid_Val_3_8_11 = sum(Val_Hybrid_3_8_11)/12
Av_Hybrid_Val_3_8_11.to_csv(path + " " + keyword_Hybrid.replace("*","") + " " + keyword_03_08_11.replace("*","") +'.csv',
                            header=None, index = False, encoding='utf-8')
Av_Hybrid_Val_4_5_12 = sum(Val_Hybrid_4_5_12)/12
Av_Hybrid_Val_4_5_12.to_csv(path + " " + keyword_Hybrid.replace("*","") + " " + keyword_04_05_12.replace("*","") +'.csv',
                            header=None, index = False, encoding='utf-8')
```

In [ ]:

```python
# A function to arrange rocks in order - order set by the original Nosofsky image folders rock order

def arrangeValidationInRockOrder(all_dfs):
    Granite = pd.DataFrame()
    for df in all_dfs:
        a = df.iloc[0:3]
        Granite = pd.concat([Granite, a], axis=0,ignore_index=True)

    Obsidian = pd.DataFrame()
    for df in all_dfs:
        a = df.iloc[3:6]
        Obsidian = pd.concat([Obsidian, a], axis=0,ignore_index=True)

    Pegmatite = pd.DataFrame()
    for df in all_dfs:
        a = df.iloc[6:9]
        Pegmatite = pd.concat([Pegmatite, a], axis=0,ignore_index=True)

    Pumice = pd.DataFrame()
    for df in all_dfs:
        a = df.iloc[9:12]
        Pumice = pd.concat([Pumice, a], axis=0,ignore_index=True)

    Gneiss = pd.DataFrame()
    for df in all_dfs:
        a = df.iloc[12:15]
        Gneiss = pd.concat([Gneiss, a], axis=0,ignore_index=True)

    Marble = pd.DataFrame()
    for df in all_dfs:
        a = df.iloc[15:18]
        Marble = pd.concat([Marble, a], axis=0,ignore_index=True)

    Slate = pd.DataFrame()
    for df in all_dfs:
        a = df.iloc[18:21]
        Slate = pd.concat([Slate, a], axis=0,ignore_index=True)

    Breccia = pd.DataFrame()
    for df in all_dfs:
        a = df.iloc[21:24]
        Breccia = pd.concat([Breccia, a], axis=0,ignore_index=True)

    Conglomerate = pd.DataFrame()
    for df in all_dfs:
        a = df.iloc[24:27]
        Conglomerate = pd.concat([Conglomerate, a], axis=0,ignore_index=True)

    Sandstone = pd.DataFrame()
    for df in all_dfs:
        a = df.iloc[27:]
        Sandstone = pd.concat([Sandstone, a], axis=0,ignore_index=True)

    return Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone
```

In [ ]:

```python
# A function to append four data sets in user defined order

def appendAverageDfs(valSet1, valSet2, valSet3, valSet4):
    all_Arrange = []
    all_Arrange.append(valSet1)
    all_Arrange.append(valSet2)
    all_Arrange.append(valSet3)
    all_Arrange.append(valSet4)
    return all_Arrange
```

In [ ]:

```python
# Arrangement 1
# C2
all_Ar1_C2_Dfs = appendAverageDfs(Av_C2_Val_1_2_3, Av_C2_Val_4_5_6, Av_C2_Val_7_8_9, Av_C2_Val_10_11_12)

[Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone] = arrangeValidationInRockOrder(all_Ar1_C2_Dfs)

all_Rocks_Ar1_C2 = pd.concat([Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone], axis=0, ignore_index=True)
all_Rocks_Ar1_C2.to_csv(path + '_all_Rocks_Ar1_C2.csv', header=None, index = False, encoding='utf-8')
```

In [ ]:

```python
# Arrangement 1
# Hybrid
all_Ar1_Hybrid_Dfs = appendAverageDfs(Av_Hybrid_Val_1_2_3, Av_Hybrid_Val_4_5_6, Av_Hybrid_Val_7_8_9, Av_Hybrid_Val_10_11_12)

[Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone] = arrangeValidationInRockOrder(all_Ar1_Hybrid_Dfs)

all_Rocks_Ar1_Hybrid = pd.concat([Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone], axis=0, ignore_index=True)
all_Rocks_Ar1_Hybrid.to_csv(path + '_all_Rocks_Ar1_Hybrid.csv', header=None, index = False, encoding='utf-8')
```

In [ ]:

```python
# Function to rearrange data based on new index variables for arrangement 2 & 3
def setSortIndex(rockName, index):
    rockName = rockName.set_index([index])
    rockName = rockName.sort_index()
    return rockName
```

In [ ]:
```
# Arrangement 2

# Set index as a list variable
Ar2_index = [1, 4, 7, 5, 8, 10, 2, 9, 11, 3, 6, 12 ]

# C2
all_Ar2_C2_Dfs = appendAverageDfs(Av_C2_Val_1_4_7, Av_C2_Val_5_8_10, Av_C2_Val_2_9_11, Av_C2_Val_3_6_12)

[Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone] = arrangeValidationInRockOrder(all_Ar2_C2_Dfs)

Granite = setSortIndex(Granite, Ar2_index)
Obsidian = setSortIndex(Obsidian, Ar2_index)
Pegmatite = setSortIndex(Pegmatite, Ar2_index)
Pumice = setSortIndex(Pumice, Ar2_index)
Gneiss = setSortIndex(Gneiss, Ar2_index)
Marble = setSortIndex(Marble, Ar2_index)
Slate = setSortIndex(Slate, Ar2_index)
Breccia = setSortIndex(Breccia, Ar2_index)
Conglomerate = setSortIndex(Conglomerate, Ar2_index)
Sandstone = setSortIndex(Sandstone, Ar2_index)

all_Rocks_Ar2_C2 = pd.concat([Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone], axis=0, ignore_index=True)
all_Rocks_Ar2_C2.to_csv(path + '_all_Rocks_Ar2_C2.csv', header=None, index = False, encoding='utf-8')
```

In [ ]:
```
# Hybrid
all_Ar2_Hybrid_Dfs = appendAverageDfs(Av_Hybrid_Val_1_4_7, Av_Hybrid_Val_5_8_10, Av_Hybrid_Val_2_9_11, Av_Hybrid_Val_3_6_12)

[Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone] = arrangeValidationInRockOrder(all_Ar2_Hybrid_Dfs)

Granite = setSortIndex(Granite, Ar2_index)
Obsidian = setSortIndex(Obsidian, Ar2_index)
Pegmatite = setSortIndex(Pegmatite, Ar2_index)
Pumice = setSortIndex(Pumice, Ar2_index)
Gneiss = setSortIndex(Gneiss, Ar2_index)
Marble = setSortIndex(Marble, Ar2_index)
Slate = setSortIndex(Slate, Ar2_index)
Breccia = setSortIndex(Breccia, Ar2_index)
Conglomerate = setSortIndex(Conglomerate, Ar2_index)
Sandstone = setSortIndex(Sandstone, Ar2_index)

all_Rocks_Ar2_Hybrid = pd.concat([Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone], axis=0, ignore_index=True)
all_Rocks_Ar2_Hybrid.to_csv(path + '_all_Rocks_Ar2_Hybrid.csv', header=None, index = False, encoding='utf-8')
```

In [ ]:
```
# Arrangement 3

# C2
all_Ar3_C2_Dfs = appendAverageDfs(Av_C2_Val_1_6_9, Av_C2_Val_2_7_10, Av_C2_Val_3_8_11, Av_C2_Val_4_5_12)

[Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone] = arrangeValidationInRockOrder(all_Ar3_C2_Dfs)

# Set index as a list variable
Ar3_index = [1, 6, 9, 2, 7, 10, 3, 8, 11, 4, 5, 12]

Granite = setSortIndex(Granite, Ar3_index)
Obsidian = setSortIndex(Obsidian, Ar3_index)
Pegmatite = setSortIndex(Pegmatite, Ar3_index)
Pumice = setSortIndex(Pumice, Ar3_index)
Gneiss = setSortIndex(Gneiss, Ar3_index)
Marble = setSortIndex(Marble, Ar3_index)
Slate = setSortIndex(Slate, Ar3_index)
Breccia = setSortIndex(Breccia, Ar3_index)
Conglomerate = setSortIndex(Conglomerate, Ar3_index)
Sandstone = setSortIndex(Sandstone, Ar3_index)

all_Rocks_Ar3_C2 = pd.concat([Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone], axis=0, ignore_index=True)
all_Rocks_Ar3_C2.to_csv(path + '_all_Rocks_Ar3_C2.csv', header=None, index = False, encoding='utf-8')
```

In [ ]:
```
# Arrangement 3

# Hybrid
all_Ar3_Hybrid_Dfs = appendAverageDfs(Av_Hybrid_Val_1_6_9, Av_Hybrid_Val_2_7_10, Av_Hybrid_Val_3_8_11, Av_Hybrid_Val_4_5_12)

[Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone] = arrangeValidationInRockOrder(all_Ar3_Hybrid_Dfs)

Granite = setSortIndex(Granite, Ar3_index)
Obsidian = setSortIndex(Obsidian, Ar3_index)
Pegmatite = setSortIndex(Pegmatite, Ar3_index)
Pumice = setSortIndex(Pumice, Ar3_index)
Gneiss = setSortIndex(Gneiss, Ar3_index)
Marble = setSortIndex(Marble, Ar3_index)
Slate = setSortIndex(Slate, Ar3_index)
Breccia = setSortIndex(Breccia, Ar3_index)
Conglomerate = setSortIndex(Conglomerate, Ar3_index)
Sandstone = setSortIndex(Sandstone, Ar3_index)

all_Rocks_Ar3_Hybrid = pd.concat([Granite,Obsidian,Pegmatite,Pumice,Gneiss,Marble,Slate,Breccia,Conglomerate,Sandstone], axis=0, ignore_index=True)
all_Rocks_Ar3_Hybrid.to_csv(path + '_all_Rocks_Ar3_Hybrid.csv', header=None, index = False, encoding='utf-8')
```

In [ ]:
```
# Add the three arrangements of data together and divide by 3

# C2
All_C2_Arrangements = pd.DataFrame()
All_C2_Arrangements = all_Rocks_Ar1_C2.add(All_C2_Arrangements, fill_value=0)
All_C2_Arrangements = all_Rocks_Ar2_C2.add(All_C2_Arrangements, fill_value=0)
All_C2_Arrangements = all_Rocks_Ar3_C2.add(All_C2_Arrangements, fill_value=0)

Av_C2_Arrangements = All_C2_Arrangements/3
Av_C2_Arrangements.to_csv(path+"_Av_C2_Arrangements.csv", header=None, index = False, encoding='utf-8')
```

In [ ]:
```python
# Add the three arrangements of data together and divide by 3

# Hybrid
All_Hybrid_Arrangements = pd.DataFrame()
All_Hybrid_Arrangements = all_Rocks_Ar1_Hybrid.add(All_Hybrid_Arrangements, fill_value=0)
All_Hybrid_Arrangements = all_Rocks_Ar2_Hybrid.add(All_Hybrid_Arrangements, fill_value=0)
All_Hybrid_Arrangements = all_Rocks_Ar3_Hybrid.add(All_Hybrid_Arrangements, fill_value=0)

Av_Hybrid_Arrangements = All_Hybrid_Arrangements/3
Av_Hybrid_Arrangements.to_csv(path+"_Av_Hybrid_Arrangements.csv", header=None, index = False, encoding='utf-8')
```

In [ ]:
```python
### Don't forget to change! ###

root = "C:/Users/c21012241/Dropbox"

### 13 Features
pathExpert = root +"/expertRatings/expertRatings - Binary Crystals/Binary Crystals - 13 Features/\
Ratings transformed - for use in matlab visual/expertRatings.csv"
#pathExpert = root +"/expertRatings/expertRatings - Continuous Crystals/Continuous Crystals - 13 Features/expertRatings.csv"

### 12 Features

#pathExpert = root +"/expertRatings/expertRatings - Binary Crystals/Binary Crystals - 12 Features/expertRatings.csv"
#pathExpert = root +"/expertRatings/expertRatings - Continuous Crystals/Continuous Crystals - 12 Features/expertRatings.csv"


### Re-rated 13 expert features
#pathExpert = root +"/XAI_Feature_Anomyly/XAI_Feature_Anomyly/expertFeatures13Binary - Correlation Plot Set - Transformed.csv"

expertRatings = pd.read_csv(pathExpert, header = None)
```

In [ ]:
```python
expertHybridCorrelation = expertRatings.corrwith(Av_Hybrid_Arrangements)
print(expertHybridCorrelation)
expertHybridCorrelation.to_csv(path+"_expertHybridCorrelation.csv", header=None, index = False, encoding='utf-8')
```

In [ ]:
```python
expertC2Correlation = expertRatings.corrwith(Av_C2_Arrangements)
print(expertC2Correlation)
expertC2Correlation.to_csv(path+"_expertC2Correlation.csv", header=None, index = False, encoding='utf-8')
```

In [ ]:
```python
hybridC2Correlation = Av_Hybrid_Arrangements.corrwith(Av_C2_Arrangements)
print(hybridC2Correlation)
hybridC2Correlation.to_csv(path+"_hybridC2Correlation.csv", header=None, index = False, encoding='utf-8')
```

In [ ]:

```
import plotly.graph_objects as go
import kaleido

# 13 Features

features = ['Average Grainsize','Roughness', 'Presence of Foliation', 'Presence of Banding', 'Heterogeneity of Grainsize',
'Lightness of Colour', 'Heterogeneity of Hue', 'Heterogeneity of Brightness', 'Volume of Vesicles', 'Glasslike Texture',
'Angular Clasts', 'Rounded Clasts', 'Presence of Crystals']

# 12 Features

#features = ['Average Grainsize','Roughness', 'Presence of Foliation', 'Presence of Banding', 'Heterogeneity of Grainsize',
#'Lightness of Colour', 'Heterogeneity of Hue', 'Volume of Vesicles', 'Glasslike Texture',
#'Angular Clasts', 'Rounded Clasts', 'Presence of Crystals']

fig = go.Figure()
fig.add_trace(go.Bar(
    x=features,
    y=expertHybridCorrelation,
    name='Expert vs Hybrid',
    marker_color="rgb(253,174,97)"
))
fig.add_trace(go.Bar(
    x=features,
    y=expertC2Correlation,
    name='Expert vs C2',
    marker_color="rgb(178,223,138)"
))
fig.add_trace(go.Bar(
    x=features,
    y=hybridC2Correlation,
    name='Hybrid vs C2',
    marker_color="rgb(116,173,209)"
))

title="13 Feature Concepts - Binary Crystal Rating - Correlation: C2 LR10^-3 E200 MB1025 - H LR10^-3 E50 MB1024 - 12 of 12"

fig.update_layout(barmode='group',
                    xaxis_tickangle=-45,
                    plot_bgcolor="#fff",
                    title=title,
                    xaxis_title="Rock Feature",
                    yaxis_title="Pearson Correlation Coefficient",
                font=dict(
                    family="Helvetica",
                    size=9,
                    color="Black"))
fig.update_yaxes(showgrid=True, gridwidth=1, gridcolor='LightPink')
fig.update_layout(title_pad_l=400,
                    title_pad_r=400,
                    title={
                    'text': title,
                    'y':0.9,
                    'x':0.5,
                    'xanchor': 'center',
                    'yanchor': 'top'})

fig.show()

fig.write_image(path + "/" + "Correlation - C2 vs Hybrid Networks.png")
```