

Common Errors in CW2

Results presentation

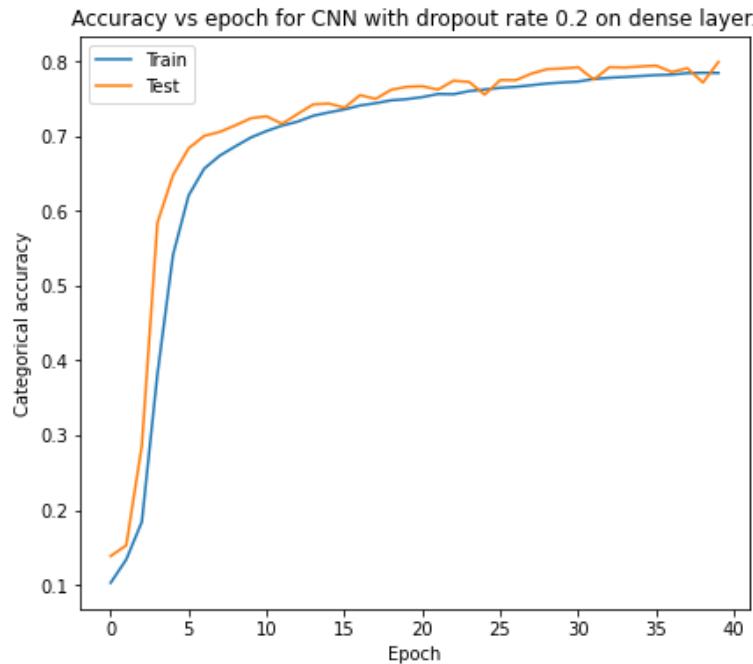
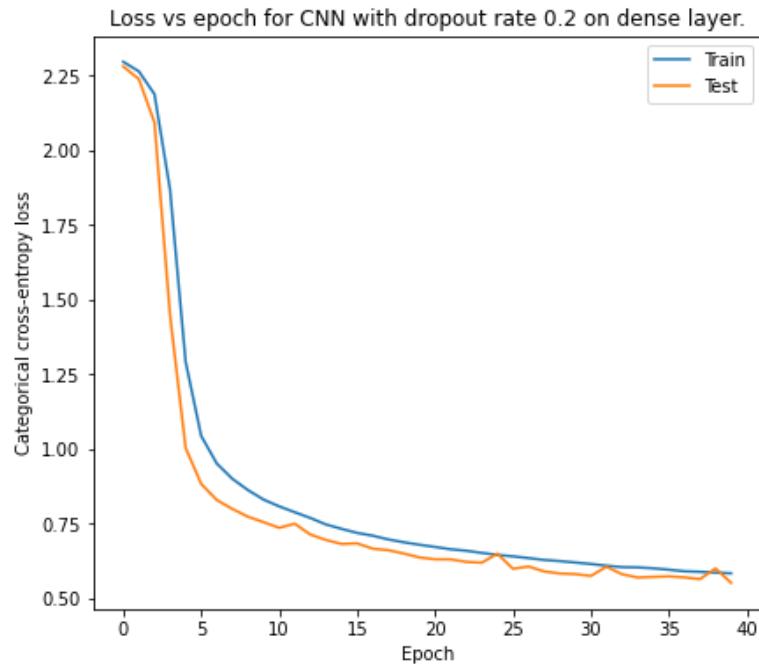
Common errors in presenting the results:

- Missing/incorrect basic plotting elements:
 - plot labels
 - axis labels
 - plot legends
- For plots made for comparison you should:
 - Place side-by-side plots that you are comparing, otherwise comparisons and analysis can be very inconvenient.
 - Efficiently use different colors/styles with employing legends.

Common Errors in CW2

Results presentation

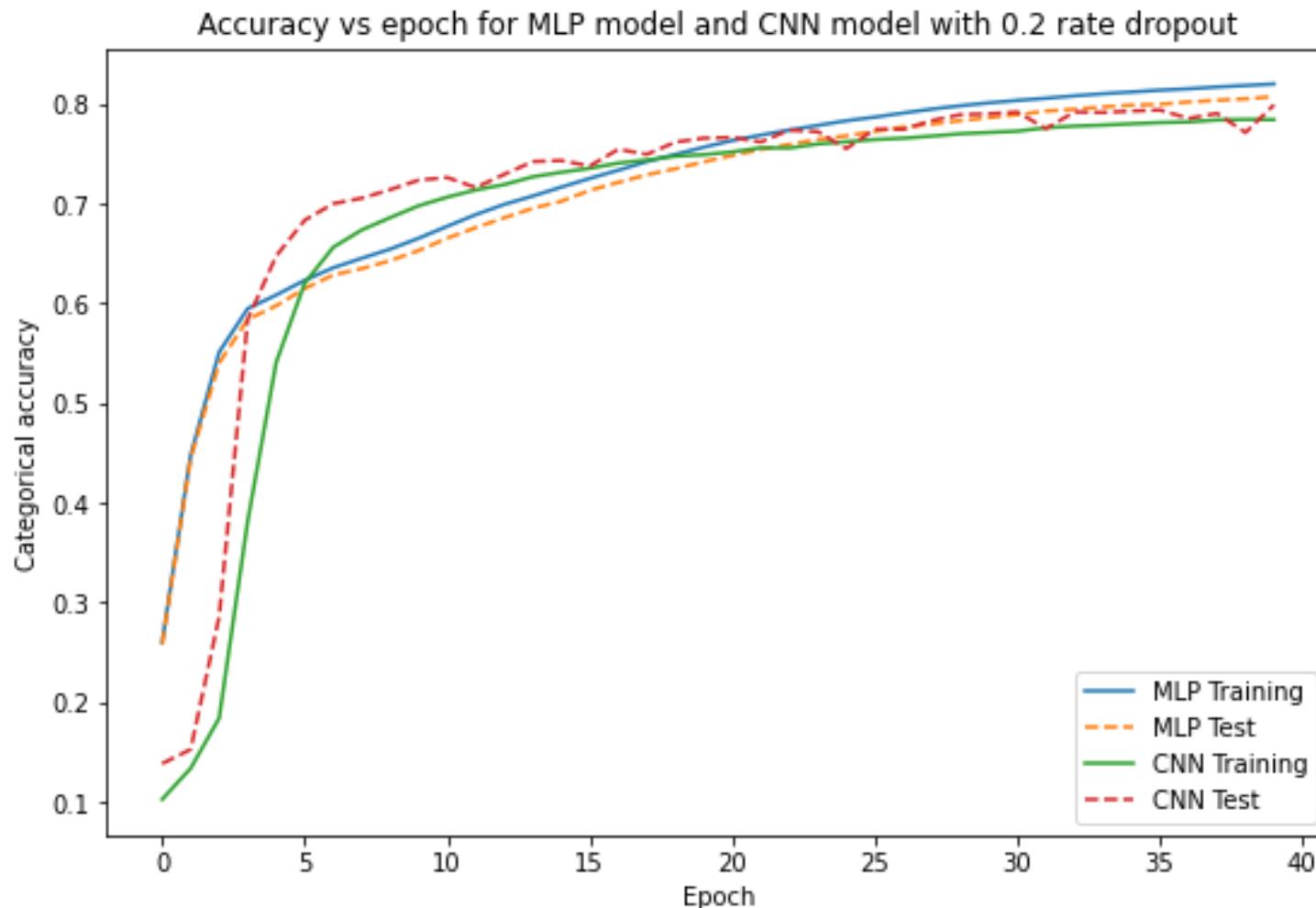
Good examples in presenting the results



Common Errors in CW2

Results presentation

Good examples in presenting the results



Common Errors in CW2

Results presentation

- For `networkx` plots, you should fix the node positions for each plot for the same dataset, otherwise comparison is difficult.

Good example

```
G = nx.Graph(A)
# 'pos' is obtained once, then used multiple times.
pos = nx.spring_layout(G, seed=1)
...
plt.title("Degree centrality")
nx.draw(G, pos, node_color=degree_centrality)
...
plt.title("eigenvector centrality")
nx.draw(G, pos, node_color=eigenvector_centrality)
...
plt.title("Pagerank centrality")
nx.draw(G, pos, node_color=pr_centrality)
```

Common Errors in CW2

Results presentation

- *Although not penalised*, if you are discussing a network in terms of vertices, it would better to visualize the network with labels.

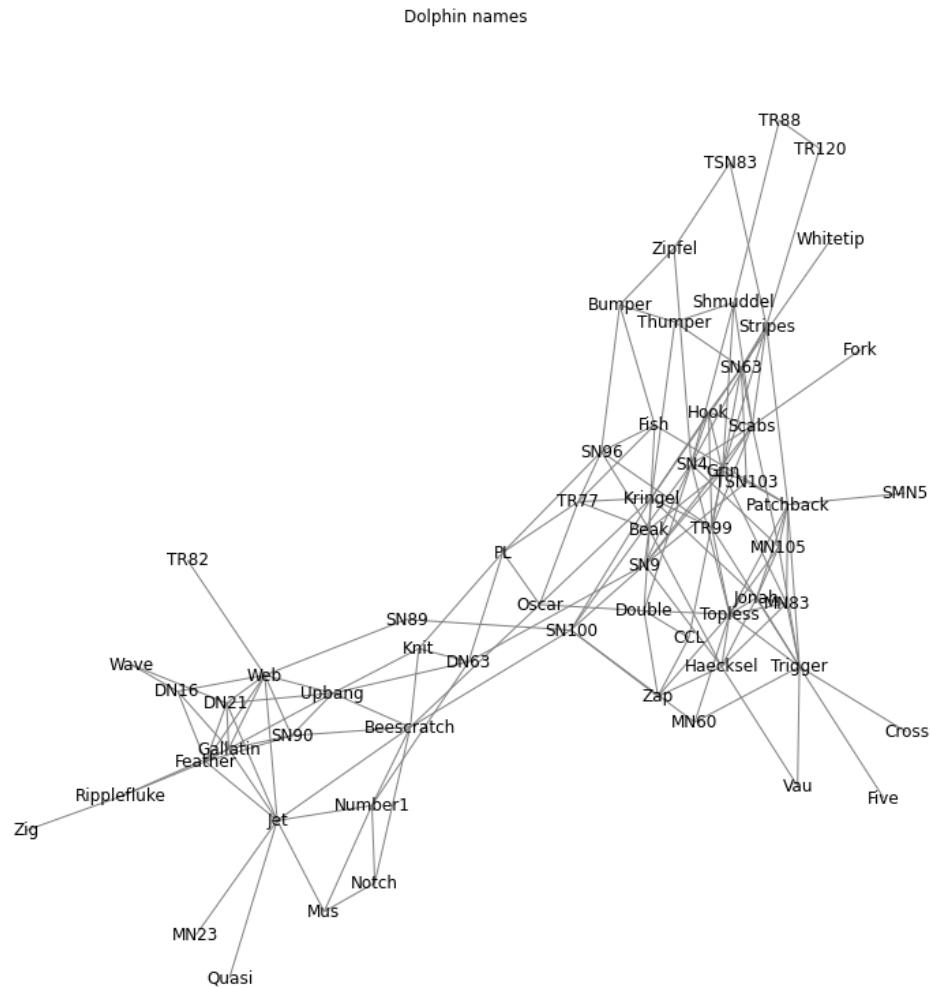
Good example

```
G_labelled = nx.relabel_nodes(G, dict(zip(range(62), names)))
plt.title("Dolphin names")
pos_labelled = nx.spring_layout(G_labelled, seed=1)
nx.draw(G_labelled, pos_labelled, with_labels=True, node_size=0)
```

Common Errors in CW2

Results presentation

Output



Common Errors in CW2

Evaluation of NumPy implementation for Task 1.1

- In Task 1.1, you were required to provide a NumPy implementation for:
 - MLP
 - Backpropagation
 - Stochastic gradient descent
 - Loss and training functions
- You lose the mark of any part in these cases:
 - no attempt.
 - missing lines (incomplete, not-working implementation)
 - your program raises error messages
 - more than one bug in your implementation.
- More marks can be lost for badly written code.

Common Errors in CW2

Task 1.2

- Should select best dropout value based on the **final accuracy**, not mean accuracy over epochs.
- Incorrect number of layers.
- Should not apply activation function on the input.

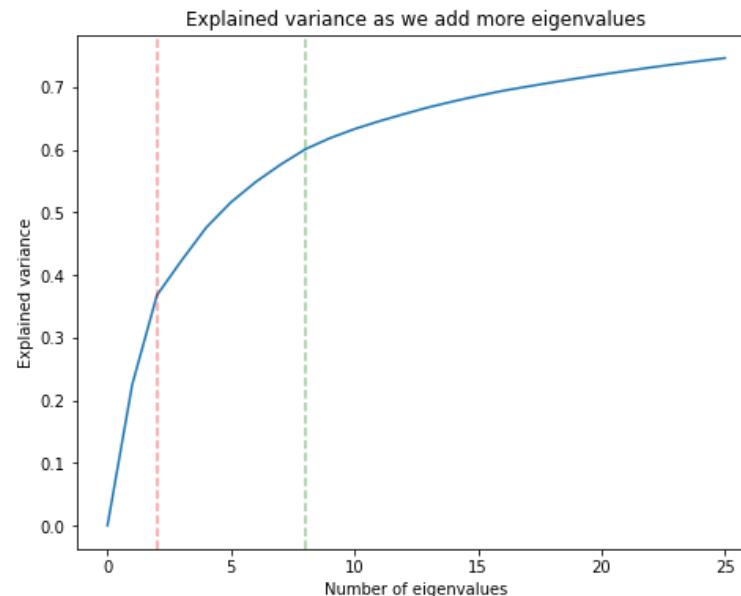
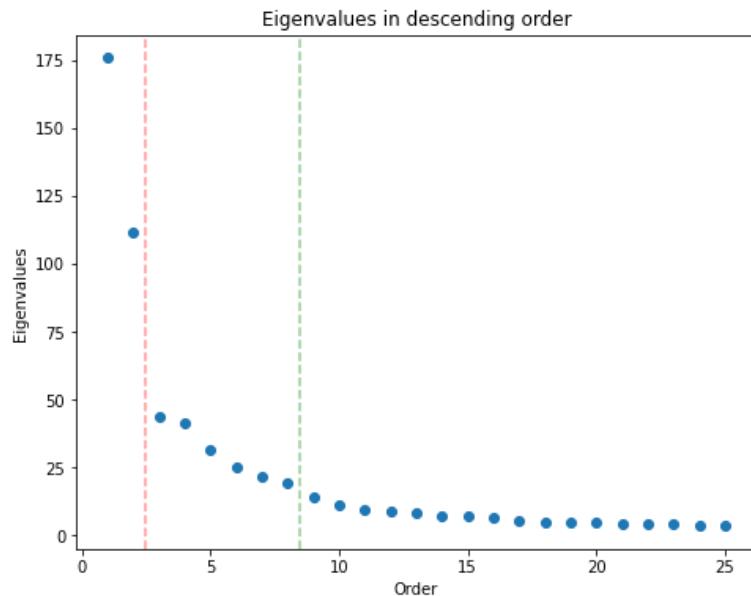
```
model_cnn = Sequential(  
    [  
        InputLayer(input_shape=(28, 28, 1), name="Input"),  
        LeakyReLU(alpha=0.01), # <- Incorrect.  
        Conv2D(8, (3, 3), activation=LeakyReLU(alpha=0.01)),  
        Conv2D(8, (3, 3), activation=LeakyReLU(alpha=0.01)),  
        Conv2D(16, (3, 3), activation=LeakyReLU(alpha=0.01)),  
        Conv2D(16, (3, 3), activation=LeakyReLU(alpha=0.01)),  
        MaxPool2D((2, 2)),  
        Flatten(),  
        Dense(64, activation=LeakyReLU(alpha=0.01)),  
        Dense(10, activation="softmax"),  
    ]  
)
```

Common Errors in CW2

Task 2.1

- Explained variance should be computed **cumulatively**.

Good example



Common Errors in CW2

Task 2.1

- Ground truth classes should not be used to determine optimal k because k-means clustering is an unsupervised algorithm.
- Should comment clearly on the case of k = 10 in k-means clustering.

Task 2.2

- Should plot *silhouette* score over the number of clusters.
- Should estimate optimal number of clusters.

Common Errors in CW2

Task 2.3

- Should discuss the meaning of a small Fiedler eigenvalue.
- Should interpret and discuss the first Eigenvector (where Eigenvalue=0).
- Most students either have not commented on the first vector or mentioned that it is a vector of ones (incorrect for normalized Laplacian).

Correct student answer

Smallest eigenvector(v_1)

Our smallest eigenvalue = 0. Note that for a given Laplacian of any undirected graph, a vector of ones is an eigenvector with eigenvalue 0.

$$L\mathbf{1} = 0$$

We will assume that the matrix is connected and we can verify when plotting the graph later on. Thus, we obtain the following expression for v_1 , where λ is a scalar.

$$LD^{-\frac{1}{2}}v_1 = 0$$

$$D^{-\frac{1}{2}}v_1 = \lambda\mathbf{1}$$

We can think of v_1 as a vector of ones under the change of basis. We can verify the expression above by calculating and writing out the first few terms of $D^{-\frac{1}{2}}v_1$

```
(mod_degree@eigenvecs.T[0])[:8]
```

```
array([-0.05607722, -0.05607722, -0.05607722, -0.05607722, -0.05607722,
       -0.05607722, -0.05607722, -0.05607722])
```

Common Errors in CW2

Task 2.3

- The Fiedler eigenvector needs to be binarised for determining the bi-partition.
- Should implement error tolerance in the iterative computation of PageRank.
- The degree centrality needs to be normalised by the number of edges ($2E$).
- Should identify nodes that are ‘central’ across all three centrality measures.
- (No deduction) Pair-plots or Pearson correlation are not sufficient methods to infer similarity in ranking, Spearman correlation can be used instead.

Common Errors in CW3

Task 3.1

- (No deduction) You should rescale to [0, 1] without standardisation before.

Task 3.2

- Link your discussion about centrality to each centrality metric, because you obtain different results from each.