# Report for Project B: Classifying Images

## Problem 1: MLP Classifier

### 1A: Dataset Exploration

**Table 1A: Dataset Analysis**

| Class | # of Training Data | # of Validation Data |
|---|---|---|
| dress | 400 | 100 |
| pullover | 100 | 100 |
| top | 1 | 100 |
| trouser | 1 | 100 |
| sandal | 800 | 100 |
| sneaker | 800 | 100 |

**Caption**: The number of training data for the classes 'top' and 'trouser' is very low. This will be a major challenge when training the model because it will be trained on very limited data for images in both these classes. This applies to the pullover class as well, which is higher in number but still low compared to other classes in the training data. This means that there will be few true predictions for these classes and many more false predictions for the other classes in the dataset. Classifiers from this training dataset cannot be aptly used to accurately classify images in the classes top, trouser and pullover.

### 1B: Model Development (training & hyperparameter selection)

We did not perform any preprocessing because the data set was sufficiently processed for training with any adjustments we would potentially make. These include trimming images to be 28x28, and scaling all the pixel values in the images to be within the 0-255 interval. For optimizing our hyperparameter search, we decided to use balanced accuracy. It is a great metric for the classification task because it evaluates accuracy of classification on datasets based on the number of true positives. In addition to that, it is clearly stated in the homework details that our model will be tested based on this metric. In terms of MLP's architecture, we decided to implement an MLP with 100 neurons, 1 hidden layer, and the 'relu' activation function for the hidden layer. The reasoning behind this setup was because we didn't want to cause extreme computations with very high neuron numbers, and we observed that 'relu' performs quite well for our model. As for the MLP's optimization algorithm, we used 'adam' (the default for sklearn MLP model) because it works well with datasets that are larger in quantity. While 'lbfgs' would also have been a good solver for a training set of this size, using adam would allow us to modify and increase the size of the training data without making major changes to the model configuration in further problems.

Find the range for all the parameters we used in our Grid Search:

1. Batch size: 32, 64, 128
2. Learning Rate: adaptive, constant
3. Alpha: logspace(-4, 4, 9)

Our grid search showed that the best configuration for this model was an alpha (L2 Regularization parameter) of 0.01, batch size 32 and ‘constant’ learning rate. This is reflected in the table below.

**Table 1B: Hyperparameter Configurations vs. Validation and Test Scores**

| *Fit time* | *alpha* | *Batch size* | *Learning rate* | *Validation score* | *Train score* |
|---|---|---|---|---|---|
| 6.456154823 | 0.0001 | 32 | constant | 0.643 | 0.800 |
| 6.999386072 | 0.001 | 32 | constant | 0.673 | 0.994 |
| 6.036937237 | 0.01 | 32 | constant | 0.806 | 0.987 |
| 4.906260014 | 10 | 32 | constant | 0.642 | 0.645 |
| 3.73647809 | 1000 | 32 | constant | 0.636 | 0.635 |

**Caption:** The table has a variety of configurations including underfitting like in 5th row and overfitting like in 2nd row. It is evident that too big alpha values lead to under-fit models and too little alpha values cause over-fit models. The reason for this pattern must be due to the regularization effect on the models.  Our major takeaway is that the optimal hyperparameter configuration includes an alpha value that is in the middle like 0.01.

### 1C: Model Analysis

**Confusion Matrix on Validation Set, Balanced Accuracy=0.806**

| | dress | pullover | top | trouser | sandal | sneaker |
|---|---|---|---|---|---|---|
| dress | 98 | 2 | 0 | 0 | 0 | 0 |
| pullover | 6 | 94 | 0 | 0 | 0 | 0 |
| top | 67 | 33 | 0 | 0 | 0 | 0 |
| trouser | 64 | 5 | 0 | 31 | 0 | 0 |
| sandal | 0 | 0 | 0 | 0 | 92 | 8 |
| sneaker | 0 | 0 | 0 | 0 | 2 | 98 |

**Caption:** The model performs poorly on images for tops and trousers. This is evident from the fact that out of no tops and only 31 trousers were accurately identified respectively (true predictions). The primary reason behind this is likely the lack of training data for both these classes, which means the model has limited information when classifying data that actually belongs to these sets.

### 1D: Training Set Modification

The original dataset lacks examples for tops and trousers, which leads to poor prediction results for images in these classes. For this reason, we decided to duplicate the images in the dataset so that each class would have the exact same number of images (800 each). To achieve this, we duplicated all dress images two times (400 x 2 = 800), all pullover images 8 times (100 x 8 = 800), and both top and trouser images 800 times (1 x 800 = 800). The reasoning behind this is that having a balanced data set prevents the model from being biased towards majority classes. So, ensuring that the number of images for each class in the training data is the same should lead the model to make more accurate predictions since the model will not be biased towards the classes with higher number of training data. However, we should note that the model will also essentially be trained on 800 replications of the same image so the training data for those classes will not capture a diverse range of images and have much less reliable predictions compared to its counterparts with less or non-duplicate images.

**Table 1E: Hyperparameter Configurations vs. Validation and Test Scores**

| model fit time | alpha | batch size | learning rate | test score | train score |
|---|---|---|---|---|---|
| **5.028097868** | 0.0001 | 32 | adaptive | 0.775 | 0.996 |
| **6.819789886** | 0.01 | 32 | adaptive | 0.747 | 0.996 |
| **5.655056715** | 10 | 32 | adaptive | 0.663 | 0.992 |
| **6.203808069** | 100 | 32 | adaptive | 0.727 | 0.984 |
| **3.852203846** | 10000 | 32 | adaptive | 0.833 | 0.937 |

The results of the hyperparameter search gave a MLP with a much higher configuration for alpha (10000.0), batch size of 32 and ‘adaptive’ learning rate. A higher L2 regularization parameter is useful for preventing overfitting (as can be seen in the table above). However, it is unexpected that the degree of alpha is so high, especially in comparison to the previous model which had an alpha of 0.01. We suspect that the high alpha values help with mitigating the overfitting effect that the duplicated data might have on the model. Additionally, there was a switch to ‘adaptive’ learning rate instead of constant. This can be due to needing a higher convergence rate for the model (since the dataset is larger) and for improved generalization due to duplicate images in training data.

### 1F: Modified Model Analysis

**Confusion Matrix on Validation Set, Balanced Accuracy=0.866**

| | dress | pullover | top | trouser | sandal | sneaker |
|---|---|---|---|---|---|---|
| dress | 96 | 1 | 1 | 2 | 0 | 0 |
| pullover | 4 | 96 | 0 | 0 | 0 | 0 |
| top | 25 | 10 | 64 | 0 | 2 | 1 |
| trouser | 9 | 6 | 1 | 84 | 0 | 0 |
| sandal | 0 | 0 | 0 | 0 | 84 | 16 |
| sneaker | 0 | 0 | 0 | 0 | 2 | 98 |

**Caption:** In comparison to 1C, this model is significantly improved in its predictions. There is a significant increase in accurate predictions for the top and trouser classes, rising from 0 to 64 and 31 to 84 true positive predictions respectively. This supports the modification procedure and hypothesis in 1E. Interestingly, the trouser class has improved much more than the top class, which is likely due to the unique shape of the clothing item. It is likely easier for a model to distinguish between a pullover and a trouser than a pullover and a top, for example. Interestingly, the true positive predictions for the sandal class dropped, which is likely due to the decreased model bias towards the sandal and sneaker classes.

### 1G: Leaderboard

The ultimate test set performance on our MLP model trained resulted in a balanced accuracy score of **0.832**.

This result is exactly in line with the model's performance on the validation set. The best score from the grid search was 0.833, which was with the same parameters that we used for the yhat_test predictions. Since the validation data was withheld when training the model, it makes sense that the model performs similarly to the test data as it did with validation as they are both unseen.

## Problem 2: Open-Ended Challenge

### 2A: Method Description

A major change that we made to improve the classifier's performance was data augmentation. We increased the diversity of the training dataset by applying small rotations, translations, and flips across the vertical and horizontal axis. For this process, we first duplicated the images just like in part1. Subsequently, we adopted a strategy of randomly selecting a transformation type (such as rotation or translation) and determining the degree of transformation randomly once more before applying it to the images. This meant that a variety of transformations were being applied across the data set, allowing for further generalization when applied to classifying unseen data. In order to decrease the computation time for our model, we applied transformations to only ⅓ of the randomly selected portion of the dataset, this also helped prevent overfitting the model. Following this approach, we increased the size of our dataset from

2,012 to 6,400, (through both modification of the class size and then augmentation through translations) which is more diverse and still manageable. There are multiple reasons why we believed that the randomized data augmentation that we've implemented would improve the model performance. First, providing the model with the diverse versions of the same data (such as rotated, flipped, and translated), helps the model recognize patterns and features relevant to these transformations then reducing overfitting. Secondly, it increases the size of the dataset, then adding more data for the model to train on without direct duplication, which helps mitigate chances of overfitting.

For the feature representation and classification method, we did not apply any modification to our previous model because having changes in all these aspects at the same time lead to poorer performance in our grid search. We attempted to use PCA for dimensionality reduction and switch to a random forest classifier, but in combination with the data set augmentation, this led to poorer performance in balanced accuracy.

**2B: Model Development (training and hyperparameter search)**

The hyperparameters we searched were alpha, batch size and learning rate. The table below demonstrates how the validation and training score were impacted by varying the size of alpha (L2 Regularization Parameter). We searched whether constant or adaptive learning rate would work better for generalization in the model, as both the size and variance in the data set had increased with the translation of the images. We searched for smaller values of batch size (8, 16, 32) as opposed to the higher values we searched for in the previous grid searches (32, 64, 128). This was because it was likely that the batch size would be smaller to improve the generalization of the model, and allow it to consider the transformations to the data set more significantly as well. We tested the same range of alpha, since it was quite inclusive of a large range and our previous results had very different values of alpha. We conducted a grid search using a splitter so that the validation data was used as a test set for the training data, all of which was packed together into a long array. The search optimizes balanced accuracy, as it is a strong method for testing true positive predictions in the model. In terms of MLP's architecture, we decided to implement an MLP with 100 neurons, 1 hidden layer, and the 'relu' activation function for the hidden layer. The model's training was conducted by the grid search which explored different combinations of hyperparameters to optimize balanced accuracy.

| **Model fit time** | **Alpha** | **Batch Size** | **Learning Rate** | **Validation Score** | **Train Score** |
|---:|---:|---:|---|---:|---:|
| **8.827037096** | 0.0001 | 32 | adaptive | 0.658 | 0.996 |
| **17.13297319** | 0.001 | 32 | adaptive | 0.708 | 0.999 |
| **9.559665918** | 0.01 | 32 | adaptive | 0.863 | 0.997 |
| **6.573436975** | 100 | 32 | adaptive | 0.700 | 0.978 |
| **7.73649478** | 1000 | 32 | adaptive | 0.847 | 0.961 |
| **5.221868038** | 10000 | 32 | adaptive | 0.717 | 0.947 |

**Caption:** There seems to be a definitive improvement in the model's performance according to the results from the hyperparameter search. Using a batch size of 32 seems to be working aptly, as does keeping the learning rate adaptive. The alpha value for L2 Regularization still has a significant impact, although there is much less evidence (if any) of the model being underfitted, even with varying values of the hyperparameters. This likely means that the data set has become more robust and the model tends towards overfitting on the varying values. The alpha value of 0.01 is giving optimal performance on the validation set.

### 2C: Model Analysis

**Confusion Matrix on Validation Set, Balanced Accuracy=0.946**

| | dress | pullover | top | trouser | sandal | sneaker |
|---|---|---|---|---|---|---|
| dress | 100 | 0 | 0 | 0 | 0 | 0 |
| pullover | 4 | 96 | 0 | 0 | 0 | 0 |
| top | 13 | 4 | 83 | 0 | 0 | 0 |
| trouser | 3 | 3 | 0 | 94 | 0 | 0 |
| sandal | 0 | 0 | 0 | 0 | 97 | 3 |
| sneaker | 0 | 0 | 0 | 0 | 2 | 98 |

**Caption:** In contrast to 1C and 1E, our model in part 2 predicts images across all classes with more accuracy. The model outperforms the other two especially in classifying tops and trousers, for which we suspect the reason to be the increased diversity in the number of training data after data augmentation. In addition, the accuracy in dress predictions increased relatively higher compared to the other classes. Although it’s difficult to pinpoint an exact reason for this, we suspect that it might be due to the randomization during data augmentation, which might have led to more training examples for the dress classification. Also, the results showcase that the model performance has significantly increased as evident from the fact that there is overall increased correct predictions across all classes which supports our hypothesis. This is due to the data augmentation providing the model with substantially more data to train on and capture the intricate relationships between features.

### 2D : Submit to Leaderboard and Record Test-Set Performance

The ultimate test set performance on our MLP model trained resulted in a balanced accuracy score of **0.885**.

This is better than our performance in Problem 1 and is likely due to the increased variation and robustness of the training set for the MLP because of augmenting the data through rotating, translating, and flipping the images. Additionally, we used the provided validation data to train the MLP used to predict the results for this leaderboard (by not using refit=False in the grid search), which may have also contributed to increasing the accuracy of the model.