Predicting Cytokine Responses From Images - Part 2!

Tempest Plott 2023-10-10

Problem Statement

Classification of microscopic images with machine learning has revolutionized biotechnology. However, the black-box nature of many machine learning approaches has left a gap between comfortable “knowns” of biologists and new “unknown” reasons for the classifications as generated by modeling. For example, some test drugs may cluster near control drugs and be listed as an exciting “hit” in a drug screen according to a clustering algorithm. But is the reason for the nearby clustering really biological, or is it due to an artifact or a phenotype unrelated to the biology being studied?

Cytokines have been used as a known descriptor of cellular activity for 50 years. By predicting cytokine production itself from images, rather than predicting similarity to control drugs, I aim to bridge the gap between the old-school and the new-school, adding confidence in the power and usefulness of ML approaches in drug discovery, improving throughput of wet-lab approaches and analysis methods, and reducing the cost of drug discovery.

The two questions addressed by the analysis presented in this report are -

1) Can the production of any of these 51 cytokines be predicted from 64x64 tiffs in CNNs or fine-tuned ViTb32?

and

2) Does employing data augmentation improve the CNN performance?

For this analysis, I define predictable cytokines as those whose production can be predicted with at least 0.65 precision and recall or 0.65 max validation accuracy. (In this analysis, chance would result in scores of 0.5.)

These questions require more context themselves, so please read the Part 1 project if you would like an explanation of the biology. (If you are currently wondering what on earth a cytokine is, I recommend reading Part 1.)

Data Collection

384 wells each containing 18,000 human white blood cells were plated.

51cytokines were quantified for each of these wells. Each cytokine measure was expressed as log10-fold change relative to the mean of negative control wells and median-shifted to center at 0. This procedure is important for consistent communication between scientists and was therefore performed before handing off the data for further analysis.

Images of one field of each well were reduced to 64x64 representations.

Cleaning the Data

Cleaning the features:

Pixel arrays were min/maxed to values of 0/1 to ensure that gradient descent steps would work appropriately across all images. (ie, it was ensured that brightness outliers would not be a detriment to learning.)

The dimensionality of the arrays was expanded to match tensorflow expectations. Later, the arrays were also expanded from one channel to three by repeating the information three times, to match the RGB format expectations of the ViT model.

Proper data types (numeric) were also enforced at this stage.

Cleaning the labels:

In the previous project, Each of 51 cytokine distributions was transformed into two bins with KBinsDiscretizer and labeled with 1 or 0 to indicate each well as either being a producer (1) or a non-producer (0) for each cytokine. Cytokines with seven or fewer producing wells were dropped from the experiment. Proper data types (numeric) were also enforced at this stage.

Figure 1: Histogram of experiment-wide normalized MMP-9 values.

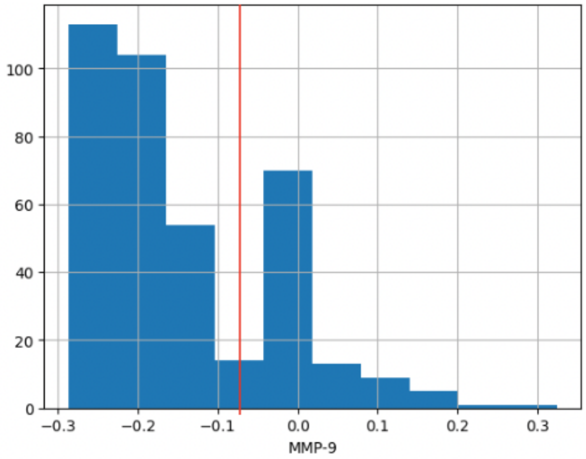


Figure 1: Example of KBinsDiscretizer splitting cytokine measurements into two bins. The Y axis is the number of wells and the X axis is the normalized cytokine value.

It is important to note that some of the cytokines are independent of each other, while others are naturally produced in concert with each other. It was thus important to separate each cytokine into its own model to see if the model can truly learn to predict just that cytokine rather than “cheating” by using the signal from another cytokine as an input. Thus, it was also important to downsample and pre-process each cytokine separately.

Exploratory Data Analysis

Features:

Augmentations were performed on the image arrays, and the arrays were shown in pseudocolor to confirm the augmentation appropriateness. Originally, rotations were employed, but these resulted in a vignette artifact. Thus, only flips were used in this work to avoid visual artifacts.

It is also noted that some tiffs are quite empty (such as the one shown below.) These empty images are not great candidates for this work, and a larger dataset should be employed in the future.

Figure 2: A successful horizontal flip, shown in pseudocolor.

Before: **After:**

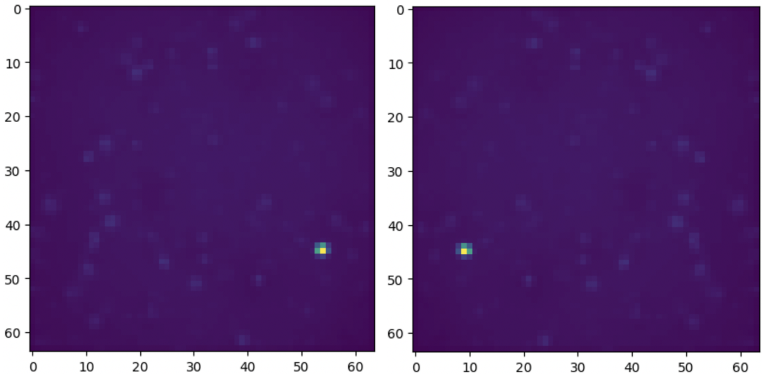


Figure 2: Well A01 has been appropriately flipped with no artifacts generated.

Labels:

The distributions of binarized cytokine production were observed. It was noted that there are many more non-producing wells than producing wells for essentially all cytokines, so a downsampling strategy was employed per-cytokine to remove non-producing wells at random to match the number of producing wells. This perfectly balanced the classes for the later modeling steps. Cytokines which had fewer than 15 wells after this downsampling process were dropped from the experiment. This left 27 cytokines to analyze.

Figure 3: Histogram of experiment-wide binarized cytokine labels.

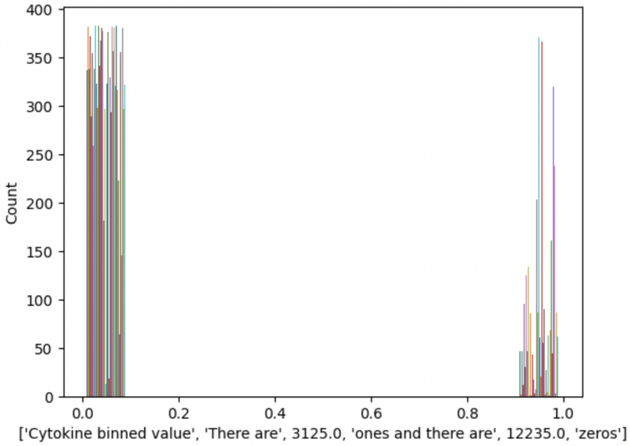


Figure 3: There was about a 4:1 ratio of wells labeled 0 to wells labeled 1 before downsampling 0s for each cytokine at random.

Data Wrangling

It was ensured that features and targets were the same length and that the format would be acceptable to the model architecture. Several lists were made to store the original pixel arrays along with various augmentation strategies. For each of these lists, labels were fetched and expanded to match the augmented feature set as appropriate, and feature and label arrays were resized as needed.

Each cytokine has its own model. The history and performance of each model was stored in a list of lists for display and summarization.

Modeling

The CNN model trained independently on each cytokine separately is as follows:

```
model = models.Sequential()
model.add(layers.Conv2D(32,(3,3), activation='relu', input_shape=(64, 64,
1)))
model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(64, (3,3), activation='relu'))
model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(128, (3,3), activation='relu'))
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(2))

model.compile(optimizer='adam',
loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
metrics=['accuracy'])
```

While the data are not particularly sparse (half of the labels are 0 and half are 1), the “Sparse Categorical Cross-entropy” loss function performed best and is not overtly inappropriate for use with this data.

The ViT model and its fine tuning, which was also applied to each cytokine separately, is as follows:

```
vit_model = vit.vit_b32(
image_size = 64,
activation = 'max',
pretrained = True,
include_top = False,
pretrained_top = False,
classes = 2)

vit_model.trainable = False
```

```
model_list.append(vit_model)
model_list.append(tf.keras.layers.Dense(256, activation = 'sigmoid'))

model_list.append(tf.keras.layers.Dense(128, activation = 'sigmoid'))

model_list.append(tf.keras.layers.Dense(64, activation = 'relu'))

model_list.append(tf.keras.layers.Dense(32, activation = 'relu'))
#final layer
model_list.append(tf.keras.layers.Dense(1, activation="sigmoid"))
#then
model = tf.keras.Sequential(model_list,name='ViT')



model.compile(optimizer='adam', loss='binary_crossentropy')
```

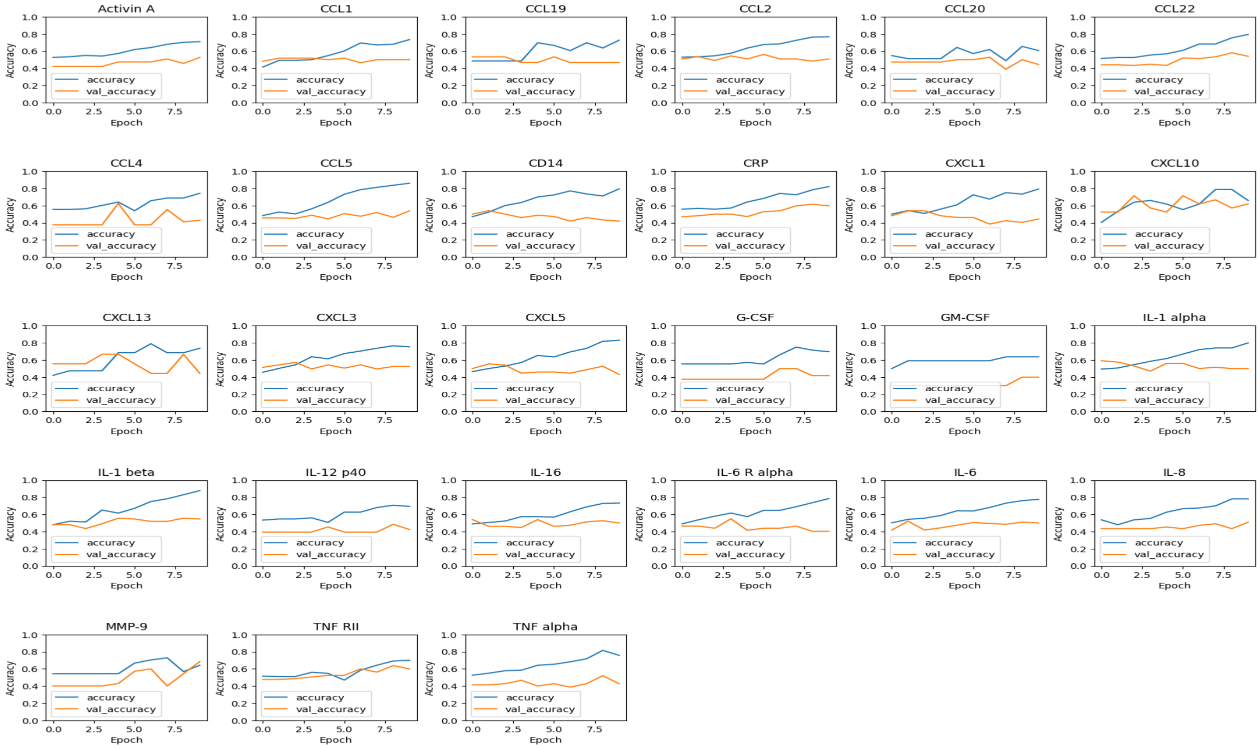
The vit_model trainability is set to false so that only the fine-tuning top layers are trained, rather than the entire generalized object detection base of the model. “Binary cross-entropy” is recommended as the default loss function for binary data, and in this case it did perform better than some other options which were used, such as mean square error. I began with sigmoid activation functions to try to retain more information from the lower layers of the model before switching to ReLu and finally, sigmoid again for the final activation to decide if the label should be “1” or not.

Results

CNNs and Overfitting

For every cytokine, test set accuracy and training set accuracy were plotted against epoch number to look for overfitting. If an epoch is reached where the test accuracy begins to get worse and diverge from the training set accuracy, that indicates that the model is becoming overfit to the training set. There were some cases of this phenomenon. There is an example below of the performance of each cytokine CNN for the dataset which contains original images and their horizontal flips. The test accuracy of CXCL1 decreases after Epoch 3 while the training accuracy continues to improve. (Cytokines in the example figure are listed in alphabetical order.) There is no particular epoch which would benefit all CNN models as a general early stopping point, so each model should be fine-tuned separately in future work.

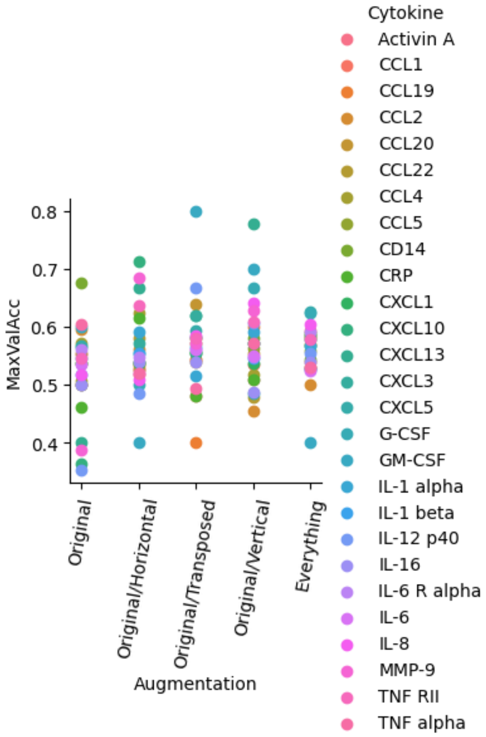
Figure 4: Accuracy Vs Epoch for all Cytokines, Original Images and Horizontal Flips



CNN Augmentation Comparisons

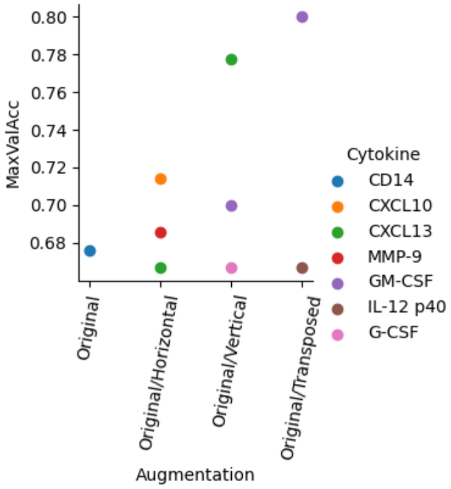
It was not the case that including all augmentations for a larger dataset resulted in better performance. There is no significant difference between which flipping strategy is employed alongside the original images, but the data consistently show that any one augmentation strategy outperforms both the original dataset and the fully augmented dataset.

Figure 5: Augmentation dataset performance for all cytokines and all augmentations



The base CNN model produced seven predictable cytokines, shown below. Interestingly, the full augmentation dataset produced no predictable cytokine.

Figure 6: Augmentation dataset performance for all predictable cytokines



ViT Performance

In a nutshell, abysmal. No cytokines are predictable with the ViT. In general, the ViT models tended to guess mostly positive or negative to rely on chance to maximize the metric rather than learn anything.

Figure 7: Example of bad ViT performance - GM-CSF, which was predictable with the CNN approach

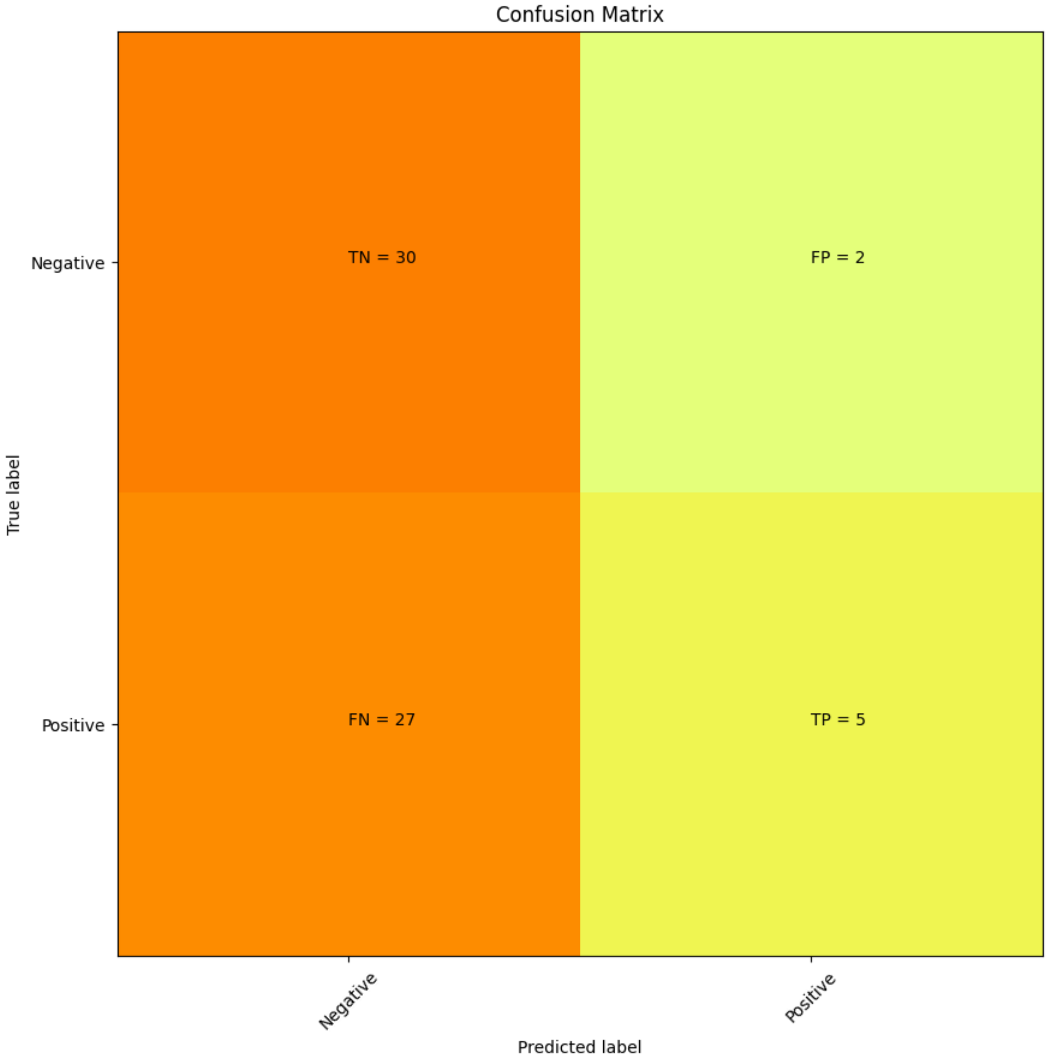


Figure 7: GM-CSF precision = 0.71, recall = 0.15

Discussion

The CNNs and ViTb32 models did not perform as well as the previous approach from part 1, which was based on image embeddings from an EfficientNetV2XLImageNet21 architecture.

The embeddings approach resulted in 10 predictable cytokines with average precision and recall of about 0.75. The CNN approach resulted in 7 predictable cytokines with average max validation accuracy of about 0.72. Only two cytokines were predictable with the original approach and the CNN approach - G-CSF and CD14. The Vit approach resulted in no predictable cytokines. This is likely due to the small sample size used in this notebook.

In the case of CNNs, more augmentation does not always lead to better performance. The embeddings used in Part 1 were created from high resolution images of the COnA channel. The Tiffs used as input here in Part 2 were low resolution. While some decrease in image quality tends to help avoid overfitting, in this case 64px X 64px was likely too low quality.

Furthermore, the CNNs and ViT used here have not been optimized for images of cells. This poor performance showed the necessity of fine-tuning ViTs on large datasets first. ViTb32 has been trained for general object detection, but microscopic images of cells is a niche enough data type that it needs much more training than I provided to it. According to a few outside resources, ViTs require larger datasets for fine-tuning than do CNNs. ViTs are often fine-tuned with about a million images, whereas I only provided 384. I was not aware of this aspect of ViTs before beginning this project, so I was glad to learn this.

Two sources discussing this issue follow:

“However, Vision Transformers typically require larger amounts of training data to achieve comparable performance to CNNs.”

“The self-attention layer of ViT lacks locality inductive bias (the notion that image pixels are locally correlated and that their correlations maps are translation-invariant). This is the reason why ViTs need more data. On the other hand, CNNs look at images through spatial sliding windows, which helps them get better results with smaller datasets.”

Future Research

There are a few things I could do to improve the performance of these models.
I should repeat this work with a larger dataset than n=384.
Each CNN for each cytokine should be tuned to avoid overfitting.
The second source linked in the above section investigates an approach to compensating for this data-hungry aspect of ViTs, which could be applied to this work.
I could also use a different pre-trained model, such as EfficientNet, rather than ViTb32.

This work can be used as 1) a proof of concept which validates the ability of the current imaging laboratory approach to capture real biological signal, 2) validation of the embeddings analysis approach which was used in Part 1, and 3) a valuable note that using newer, fancier methods does not guarantee improvement of results.

Thanks

Gratitude to Noor Hussain and Ahmed Hosny for their guidance in performing this analysis.

Two outside resources were used in this work.

Code from this post was used/adapted to create more visually appealing confusion matrices.

Code from this post was used/adapted to show learning curves and look for evidence of overfitting.