Deliverable 2

Project: Quick Draw
Team members: Anson Ng, Abigail Wolfensohn

1. Problem statement:

Our goal is to train a machine learning model to be able to predict what a person is drawing from a list of 20 prompts, based on the qualities of their real-time brushstrokes. Our work will be inspired by Google's Quick Draw game.

2. Data Preprocessing:

We are indeed still going to be working with a subset of Google's Quick Draw dataset, which features representations of 50 million doodles total, of 345 different types of objects (labels). This dataset was already preprocessed when we accessed it, with multiple formats made available to users depending on what they intend to use it for. We have decided to use the Simplified version of the dataset that comes in the form of JSON files, viewable at this link. It consists of timestamped vectors representing the manner and order of brushstrokes used to draw the doodles, which will be useful for training our model to understand when a user creates brushstrokes to draw an image. For the sake of feasibility we've chosen to narrow down these 345 categories to approximately 20 (exact ones TBD).

3. Machine learning model:

a. Framework and Tools:

Framework: recurrent neural network (RNN), most likely using Keras

Architecture:
- Input Layer: Sequence of drawing strokes from the Simplified dataset.
- RNN Layer(s): One or more layers of LSTM (Long Short-Term Memory) units. LSTM is a type of RNN that can learn and remember over long sequences and is less susceptible to the vanishing gradient problem.
- Dense Layer: Fully connected layer for classification, using softmax activation to output probabilities for each class.

Architecture Diagram:
Input Sequence (timestamped) → LSTM Layer 1 → LSTM Layer 2 (optional, based on experiment results) → Dense Layer (Softmax Activation) → Output (Class Probabilities)

b. Training/Validation/Test Splits, Regularization, and Hyper-parameters:

Dataset Split:
- 70% Training: This provides a substantial amount of data for the model to learn from.
- 15% Validation: To tune hyperparameters and check for overfitting.
- 15% Test: To evaluate the model's performance on unseen data.

Regularization:
- Dropout can be added between RNN layers to prevent overfitting.
- L1 or L2 regularization could be applied to the dense layer.

- Learning Rate: Start with a default of 0.001 and adjust based on validation performance.

Hyperparameters:
- Batch size, number of LSTM units, number of LSTM layers, dropout rate, and learning rate would be the primary hyperparameters to tune. Use a combination of manual tuning and automated methods like grid search or random search.

c. Validation Methods:

Overfitting/Underfitting:
- Monitor the training and validation loss. If training loss decreases but validation loss stagnates or increases, it's a sign of overfitting.
- Regularly save model weights and choose the model with the lowest validation loss for final testing.

Model Testing:
- Evaluate the model on the test set after final training.
- Use metrics such as accuracy, F1 score, and confusion matrix to assess performance across categories.

d. Challenges:

- Sequence Length Variability: Drawings might have variable sequence lengths. We hope to address this by padding sequences to a consistent length.
- Complexity: We expect our training to be quite resource-demanding even to run, after narrowing down our dataset from 345 to 20 labels, although this certainly helped. Running our code using an online resource (e.g. Colab) rather than locally may help to alleviate this.

4. Preliminary results:

As we have not yet formally implemented our code, we are not at a point where we are ready to present in-depth results.

5. Next steps:

Our next step is to begin officially building and training our network. This will require additional research and learning on our ends before we begin, since for both of us this is our first time working with RNNs. We also need to narrow down which 20 labels we want to use for our model. Our current plan is preliminary; therefore, if we end up with hiccups along the way we may change our plan from what is described in this Deliverable..