Hieroglyph Classification with Deep Learning

Sebastien Boxho, Miguel Handt Fueyo

Barcelona School of Economics

Introduction

Ancient Egyptian Hieroglyphs are relics of an ancient time that have fascinated many across centuries. Nowadays, millions of tourists visit historic places in Egypt every year. Interested tourists could profit from a deep learning approach that can process pictures of hieroglyphs and return their rough meaning. This would be especially impressive considering that the knowledge of hieroglyphs had been lost completely in the medieval period and was deciphered due to the discovery of the Rosetta Stone in 1799, which was inscripted with hieroglyphs and a Greek translation. Deep learning approaches can then be developed thanks to the work of Egyptologists like Sir Alan Gardiner, who created a list of common hieroglyphs with their corresponding codes (See the complete list: link)

Literature

Developments in deep learning applied to computer vision tasks can be applied to the task of associating hieroglyphs with their corresponding codes. We mostly follow the approach in Barucci et al. (2021). The authors use a publicly available dataset created by Franken and van Gemert (2013) and expand it by labelling more hieroglyphs manually. They can do this since one of the authors is an Egyptologist, but we are limited in this regard. They then try transfer learning approaches and benchmark them against their own architecture. One part of the approach that we criticize is that they limit their labels to just the 40 most frequent out of the 172 available ones in the public dataset, the reasoning being that a large part of the other labels is significantly underrepresented (between 1 to 8 samples). We argue that in order for the model to be used in practice, it needs to be able to predict as many hieroglyphs at possible. Therefore we include all of codes but restrict the validation and test sets to have only those with more than 8 samples.

Dataset

The dataset made available by Franken and van Gemert (2013) includes 4210 labelled hieroglyphs extracted from 10 grayscale pictures included in Piankoff (1955) (See Examples in Figure 1). The distribution of the top most frequent labels is highly imbalanced as shown in Figure 2. The codes N35 and M17, which stand for “water ripple” and “reed leaf” respectively, are represented comparatively frequently. The code “UNKNOWN” includes all hieroglyphs which are not legible or otherwise recognisable, therefore it has to be excluded to avoid adding random noise to the model.

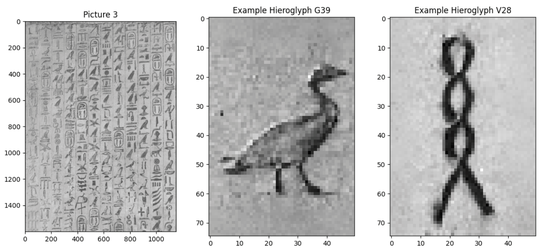

Figure 1. Example hieroglyphs out of Picture 3

The imbalance can be adressed by resampling the underrepresented classes as shown in Figure 2 with the vertical red line. All underrepresented classes have been upsampled to reach 32 samples, without downsampling of the majority classes.

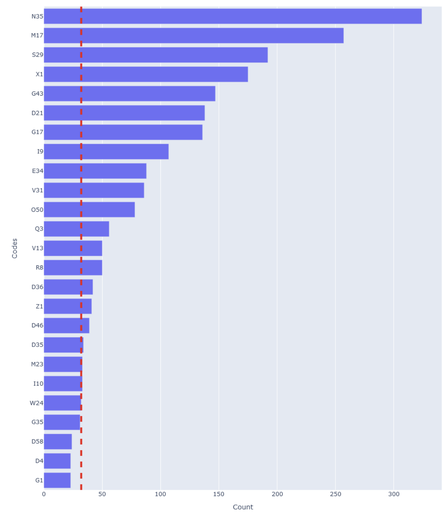

Figure 2. Original Distribution and Effect of Upsampling in Red

Model

Barucci et al. (2021) use different CNN architectures and find that their model beats the performance of other frequently used architectures (ResNet-50, Inception-v3 and Xception) even when transfer learning is being applied. We try to replicate this result by running a pretrained ResNet-50 model and benchmarking it against the personalized architecture presented in the paper, which is as follows:

The architecture consists of 6 blocks:

- The first block is the input block and consists of two identical sections, each containing:
  - a convolutional layer with 64 filters, a kernel size of 3 × 3, and a stride of 1 × 1;
  - a Batch Normalization (BN) layer, a max pooling layer (with a kernel size of 3 × 3 and a stride of 2 × 2), and a ReLU activation.
- This is followed by 4 blocks with the same layout but with an increasing number of filters (128, 128, 256, 256):
  - a separable convolutional layer with 128 filters, a kernel size of 3 × 3, and a stride of 1 × 1;
  - a BN layer and a ReLU activation;
  - another separable convolutional layer with 128 filters, a kernel size of 3 × 3, and a stride of 1 × 1;
  - a BN layer, a max pooling layer and a ReLU activation.
- The final block is the output block and consists of:
  - a separable convolutional layer with 512 filters, kernel size of 3 × 3, stride of 1 × 1;
  - a BN layer and a ReLU activation;
  - a Global Average Pooling layer;
  - a Dropout layer with a rate of 0.15;
  - a fully connected layer;
  - a Softmax layer to return probabilities for the classes.

Results

The results are shown in the table below but it should be noted that the authors fail to mention whether the metrics they present are the average metrics across all the classes or possibly a different statistic, for example a weighted average. We make the precautionary assumption that it is the average. We also compare the metrics only across the codes that are present in the authors’ model for a fair comparison and in the last row add the metrics for all the unique codes present in our test set which are significantly more (105).

Model	Accuracy	Precision	Recall	F1 Score
Barucci et al.	0.976	0.975	0.965	0.968
Replica	0.979	0.881	0.908	0.887
Replica + Resampling	0.954	0.81	0.823	0.805
ResNet-50	0.979	0.888	0.908	0.896
Replica All Codes	0.91	0.65	0.65	0.64
ResNet-50 All Codes	0.92	0.64	0.66	0.64

Table 1. Performance metrics of different models.

From the similar accuracies we note that there is a possiblity that the authors might have chosen to present weighted metrics, in which case if we were to do the same the performance of our models would be more similar to theirs in terms of metrics. We also have not been able to replicate their claim that their architecture performs better than ResNet-50 trained from scratch.

Since the model training time is short, we have been able to implement a grid search to find the best parameters. The resulting parameters were: A batch size of 60, no shuffling, a learning rate of 0.001, and no oversampling.

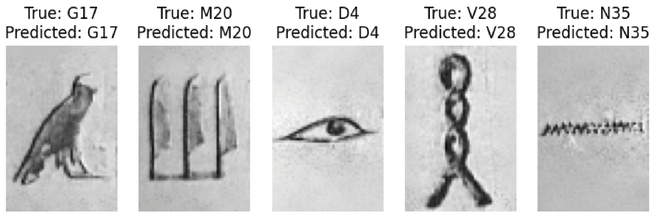

Figure 3. Example Predictions with Corresponding Codes

Conclusion

Our contribution is a model that is able to predict a larger amount of codes than the one presented in the literature. The results are decent despite a general lack of data and would need more labelling for completeness. However, we see the possibility for a promising use case where users would be able to take a picture of a hieroglyph and receive not only the code but a translation into English.

References

Barucci, A., Cucci, C., Franci, M., Loschiavo, M., and Argenti, F. (2021). A deep learning approach to ancient egyptian hieroglyphs classification. *Ieee Access*, 9:123438–123447.

Franken, M. and van Gemert, J. C. (2013). Automatic egyptian hieroglyph recognition by retrieving images as texts. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 765–768.

Piankoff, A. (1955). *The Pyramid of Unas*. Bollingen Series XL No.5.