Dysgraphia detection using machine learning

Vojtěch Formánek, 514324
FI MUNI

September 2023

1 Introduction

In recent years the interest in using machine learning to explore, screen and diagnose dysgraphia has been steadily rising. Since many of the diagnostic tools have been (in some form) digitized it has become possible to use more complex models. The method most widely used are writing tablets and pens that collect spatial and temporal data. With this aproach one can use some of the already existing diagnostic methods or theoreticaly any of the participants handwriting. As is the case with our dataset, and in most of the studies we will be referencing, the data collected are of childrens handwriting.

Dysgraphia is diagnosed using a plethora of methods, here we are interested only in data extracted from the handwriting itself. Most of the studies focus on extraction of static features from the data. Theese can range from simple features such as min, max speed of the handwriting, average tilt or on/off paper ratio . More complex metrics are used, but they do not always yield better results. A variety of models are then used to classify the data or for feature extration and dimension reduction (most commonly Random Forests, SVMs and PCA).

Though they can yield good results, there have been some attempts to explore the usecases of well known neural network architectures, mainly for classification. Notably CNNs, LSTMs (and a VAE) or, recently, a combination of both (though this has been published after the inception of this project). Here we focus on using an LSTM-based architecture, comparing it to a classification method with simple features extracted from the data and a combination of both.

2 Data collection

2.1 The original dataset

The entire dataset contains around 500 hundred participants, homewer handwriting isn't available for all of them so a subset of 490 was used. It is also extremely unbalanced. The data was collected on WACOM tablet with a pen, together collecting seven features - x, y coordinates, sample time, if the pen was touching the screen, azimuth and tilt, pressure on the tip of the pen.

A writing task was put on top of the tablet, in which the participants had to connect nine dots. First, based on an example shown before, then from memory. The dataset contains 10 tasks per participant and some aditional information, which we shall not use, and the diagnosis, a binary class.

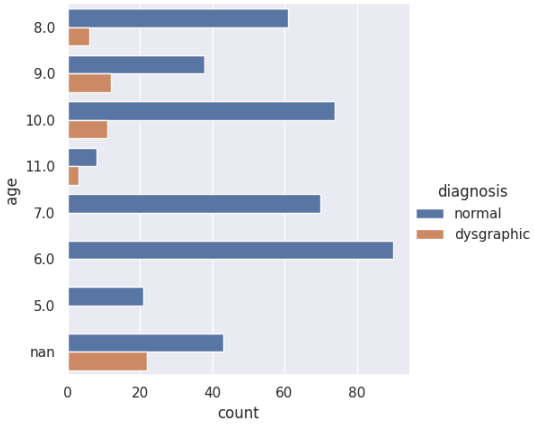

Figure 1: The dataset is indeed extremely unbalanced, also age 5-7 has no representation of dysgraphics.

2.2 Additional datasets

Drotár and Dobeš gathered a dataset with a similar WACOM tablet, thus the structure of the data was similar to the original dataset, making conversion relatively simple. On the other hand, the data collected differs. The task contains several hand written existing and made up words. The difference was not investigated further, since only subsets of the individual tasks are used to feed into the LSTM models, which are the main focus of this project.

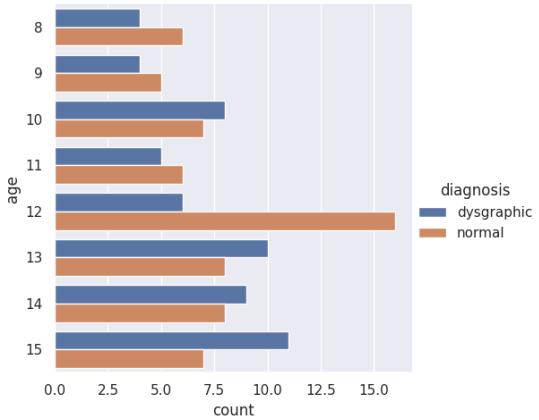

Figure 2: The additional dataset is balanced, although includes ages above our original.

3 Models

3.1 Classification of extracted features

We extracted simple features as in Drotár and Dobeš (2020). Their purpose was to train a baseline Random Forest classifier on them and compare the results to the models trained on raw data. A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting (Pedregosa et al., 2011). Random forests are popular in research partially due to the relatively high interpretability of their structure.

3.2 Classification of time-series

Other studies use LSTMs for feature extraction (Masood et al., 2023) or to approximate a score related with dysgraphia (Bublin et al., 2022). The question then arises if it is feasible, or at least possible, to train such network to predict dysgraphia directly from writing.

Firstly we constructed a baseline architecture that consists of two LSTM blocks and a single neuron fully connected layer (Masood et al., 2023). Then experimented with adding more LSTM blocks and fully connected layers on top.

And adapted an Quick, Draw! competition LSTM-architecture with 1-D convolutional layers. Consisting of two LSTM, 1-D convolution and MaxPooling blocks with two fully connected layers on top.

4 Data preprocessing

Since there is no representation of dysgraphics in the ages 5-7 and the additional dataset also does not contain any, we decided not to use them, and save them for the post-hoc analysis. Both of the datasets were split into a hold-out test set (20%), and further into validation (10%) and train (80%) sets. Splitting was done on participant level, so that there are no overlaps between the sets. Because of the small size, this prevents the models overfitting a given individuals writing.

None of the preceding studies describe preprocessing in depth, thus we compare using the raw data, filtering with Fast Fourier Transform and and a second order Butterworth filter. Using resample_poly (upsample and downsample to final frequency of 20 Hz) and butter (critical frequency of 5 and sampling freq of 15 Hz; Virtanen et al., 2020).

Originally we used Fourier-series based estimation, described above, to deal with the inconsistent sampling rate of the tablet (which ranged from 200 Hz to 1 Hz), which also removed some of the noise. A simpler solution, is to split the series into subseries in places where the sampling rate is smaller than 10 Hz. This is the solution used in the final state of the project. Even though it does add additional complexity when handling the data, it does not place too stress on the resampling.

Instead of using the absolute values for x and y we used the first order derivative (speed) in each coordinate respectively. Given that the tasks were printed out and put onto the screen in different places, the absolute values would provide unnecessary noise for the model.

Homewer we did not remove any outliers, given the scarcity of the data and the expertise required to do so.

4.1 Extraction of static features

We extracted simple features as in Drotár and Dobeš (2020). Their purpose was to train a baseline Random Forest classifier on them and compare the results to the models trained on raw data.

The total number of features extracted was 175, though some of them obviously provide no relevant data to the model (time min which is always 0). 52 were removed, thus the classifier was trained on the remaining 123. The features were scaled and combinations of oversampling, undersampling were tried out.

4.2 Raw data and the locality of dysgraphia

Since the chosen architectures require inputs of fixed length, it is neccessary to determine the correct length of the sequence. Unlike classification on inputs that have, in some form, seen the whole task. The question then arises if we can detect dysgraphia in smaller patches of a given writing sample. This potentially introduces a lot of noise into the training data, that is hard to remove.

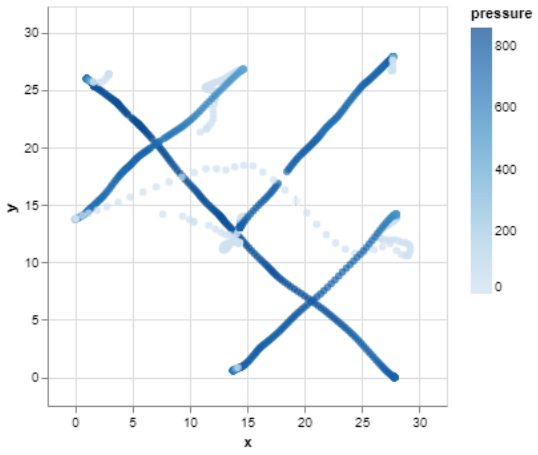

Figure 3: A reconstruction of the input. Light blue represents movements in air. Dysgraphics tend to have increased in-air movements.

Firstly we used length of 100 described in Bublin et al. (2022) then extended to 120, 240 and 480. The last is the 95 percentile of the task length in the original dataset, thus extending above that would only pad most of the sequences. The sequences were then split and used with overlaps, according to the given total length. Standardization was done locally on each sequnce, and globally, but no difference was found. Similarily replacing total time with relative from the start of the sequence, as is standard in time-series prediction, did not make a difference.

5 Evaluation

5.1 Metrics

Following the previous studies standard metrics for classification are:

$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$
$Precision = \frac{TP}{TP+FP}$
$Recall = \frac{TP}{TP+FN}$
$F1 = \frac{2*Precision*Recall}{Precision+Recall} = \frac{2*TP}{2*TP+FP+FN}$
$Sensitivity = Recall = \frac{TP}{TP+FN}$
$Specificity = \frac{TN}{FP+TN}$
AUC is calculated as the area under the $Sensitivity(1 - Specificity)$ curve.

Unweighted F1 is used, since the dataset is imbalanced the normal diagnosis would have a larger share on the score, whilst we want to mainly highlight the dysgraphic.

A baseline for all models is a dummy classifier that predicts based on the prior distribution of the classes on a balanced dataset. It's results $Accuracy = Precision = Recall = F1 = 0.5$. This is a more difficult target, given the fact that our dataset is greatly imbalanced. But since a variety of techniques, such as oversampling and class weights is used, we consider this adequate.

5.2 Classification of extracted data

If trained only on the original dataset, the Random Forest Classifier (RFC) is unable to learn and returns $F1 = 0$, classifying everyting as normal, i.e. $Recall = 0$. This is a known problem with similarly structured tasks. Even if we add the additional dataset we're unable to increase $Recall$ and thus the $F1$. Note that we used only features extracted from the handwriting to train the model.

Additionally we were able to replicate results from Drotár and Dobeš on their data only, with a 5-fold crossvalidation. Showing that classification with the features we extracted is indeed possible.

5.3 Classification of time-series data

The baseline LSTM architecture consisting of two LSTM blocks and a single output neuron, similarily to the classifiers, has trouble dealing with the original dataset only returning $F1$ scores around 0.48. Unlike the classifiers on extracted data, none of the architectures fared better on the additional dataset, or the combination. None of them were able to beat the baseline classifier $F1$ by any meaningfull margin (difference was less than 0.05). Using larger architectures with more fully connected layers led to quicker overfitting.

In most cases the loss on the validation set does not decrease after the first few epochs, signifying early overfitting. Plotting the resulting $F1$ scores follows the bell curve centered around 0.5 or the ratio of dysgraphic to normal. The AUC curve (Fig. 4) also shows that the model discriminates poorly.

Also, since the individual sequences overlap we can observe how the network treats adjacent ones. In line with the above mentioned results, it can classify the first and last of 4 sequences as dysgraphic and the inner two as normal. Even though theese have 85% overlap. Further after visual inspection, we were not able to find any meainingfull differences between, for example, false negatives and true positives, or true negatives.

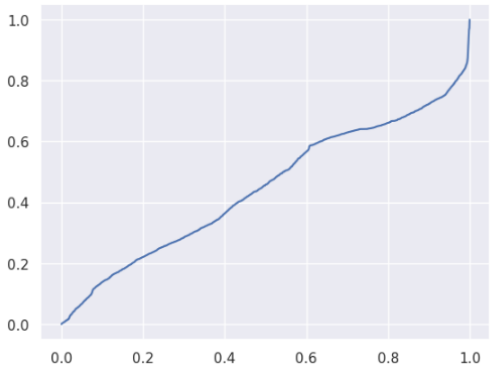

Figure 4: A typical ROC curve for the LSTM model. Topleft curve would would be close to the ideal state. Instead, going through the middle shows the weak discriminatory abilities of the models

6 Conclusion and Discussion

The networks are unable to learn on any of the three combinations of datasets. This might be to the small size, but given the relative scarcity of datasets of dygraphic handwriting, it is probably still preferable to use LSTMs as feature extractors, or in ensembles.

On the other hand studying the writing directly, not holistically, might lead to design of more expressive writing tasks and to deeper understanding of the "curve level" features of dysgraphia.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... Zheng, X. (2015). *TensorFlow: Large-scale machine learning on heterogeneous systems.* Retrieved from `https://www.tensorflow.org/` (Software available from tensorflow.org)

Bublin, M., Werner, F., Kerschbaumer, A., Korak, G., Geyer, S., Rettinger, L., & Schoenthaler, E. (2022). Automated dysgraphia detection by deep learning with sensogrip. *arXiv preprint arXiv:2210.07659*.

Drotár, P., & Dobeš, M. (2020). Dysgraphia detection through machine learning. *Scientific reports*, *10*(1), 21541.

Masood, F., Khan, W. U., Ullah, K., Khan, A., Alghamedy, F. H., & Aljuaid, H. (2023). A hybrid cnn-lstm random forest model for dysgraphia classification from hand-written characters with uniform/normal distribution. *Applied Sciences*, *13*(7), 4275.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., ... SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. doi: 10.1038/s41592-019-0686-2

Waskom, M. L. (2021). seaborn: statistical data visualization. *Journal of Open Source Software*, *6*(60), 3021. Retrieved from `https://doi.org/10.21105/joss.03021` doi: 10.21105/joss.03021