Synthetic handwriting generation

Saumya Pathak, Shipra Banga
saumyap@stanford.edu, saumyap@google.com
sbanga@stanford.edu, shiprab@google.com

Video

Stanford
Computer Science

Abstract

Mimicking another human's handwriting is tough for a human, leave alone a computer. Even the best of editors like MS Word, Google docs, do not provide fonts remotely close to human handwriting. Here, we show how Long short team memory with recurrent neural networks can be used to generate very complex, non-standard sequences such as human handwriting by predicting one data point at a time

We find that with a combination of LSTM with attention mechanism and Mixture Density networks gives us impressive results in terms of mimicking the handwriting styles from our training data set.

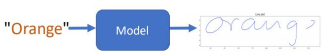
There is more improvement possible in training by handling the very prominent exploding/vanishing gradients problem in RNN. Scope of future work spans from generating handwriting in any language, or even speech synthesis by using similar granularity of data.

Stanford University

Predicting

Given a dataset of x, y coordinates of human written text on a digital surface, our model generates any given sentence in a handwriting style close to a human. We can choose between a specific style to be copied or picking a random style.

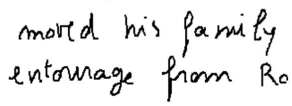
We built a model using LSTM with Mixture density networks and attention mechanism to achieve this



Data and Data processing

We used IAM Online handwriting database which contains forms of handwritten English text acquired on a whiteboard. The collected data is stored in xml-format. Apart from other data, each XML file contains a *Strokeset*, which contains a *Stroke* with its color and time information and a *Point* with x, y and time value.

Sample writing:



Models

Loss equation:

$$\mathcal{L}(\mathbf{x}) = \sum_{t=1}^{T} -\log\left(\sum_j \pi_t^j \mathcal{N}(x_{t+1}|\mu_t^j, \sigma_t^j, \rho_t^j)\right) - \begin{cases} \log e_t & \text{if}(x_{t+1})_3 = 1 \\ \log(1-e_t) & \text{otherwise} \end{cases}$$

Attention mechanism equations:

$$\phi(t,u) = \sum_{k=1}^{K} \alpha_t^k \exp\left(-\beta_t^k (\kappa_t^k - u)^2\right)$$

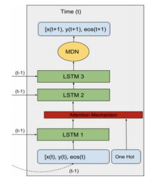
$$w_t = \sum_{u=1}^{U} \phi(t,u) c_u$$

$$(\hat{\alpha}_t, \hat{\beta}_t, \hat{\kappa}_t) = W_{h^1 p} h_t^1 + b_p$$

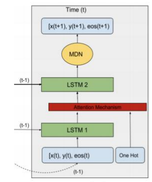
$$\alpha_t = \exp(\hat{\alpha}_t)$$

$$\beta_t = \exp(\hat{\beta}_t)$$

$$\kappa_t = \kappa_{t-1} + \exp(\hat{\kappa}_t)$$

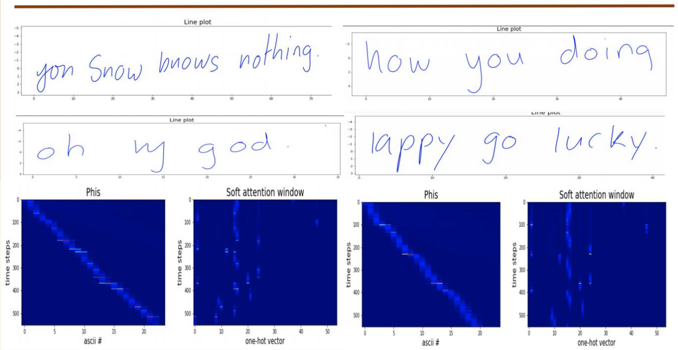


Model 1



Model 2

Results



Batch Size	#Minibatches	Dropout Rate	#Hidden Layers	Train Loss	Valid Loss
32	15	0.98	3	-4.23	-2.36
128	48	0.95	3	-2.56	-.2.34
128	48	0.95	2	-2.51	-.2.28

Features

- Picked collection of tstep points as lines.
- Removed lines with less than tstep points.
- Each point is composition of x,y coordinate and end-of-stroke boolean value
- Took offset of points from whiteboard corner coordinates.
- Scaled point values with scale factor

Future

- Train the model to generate text from any language, even symbols.
- Tweak the model to generate sensible next words.
- Improve the model to perform better for deep-in-time scenarios. Example: generating a very long sentence.
- Tackle the exploding gradient problem better

Discussion

- RNN with LSTM can learn and generate complex long-range structures using next-step prediction.
- Exploding gradients is a prominent problem while training RNNs. Exponential learning rate decay and regularization are useful to tackle this.
- Interestingly, we observed the results didn't change much even after using 2 LSTM layers instead of 3, which reduced the number of variables greatly
- The model shows potential to generate sensible next characters if it is forced to generate beyond the specified sentence.

References

1. Alex Graves. Generating Sequences with Recurrent Neural Networks. ArXiv 2013 arXiv:1308.0850
2. Realistic handwriting with tensor flow by Sam Greydanus
3. Resources on Coursera and Youtube