Realtime control of sequence generation with character based Long Short Term Memory Recurrent Neural Networks

Memo Akten, Goldsmiths University of London
Mick Grierson, Goldsmiths University of London
Huosheng Hu, University of Essex

Abstract

Recurrent Neural Networks (RNNs) — particularly Long Short Term Memory (LSTM) RNNs — are a popular and very successful model for generating sequences. However, most LSTM based sequence generation techniques are currently not interactive and do not allow continuous control of the sequence generation, let alone in a gestural or expressive manner. This research investigates methods of realtime continuous control and steering of RNN sequence generation, as well as ways of expressively controlling the output.

Introduction

Recurrent Neural Networks (RNN) are Artificial Neural Networks with recurrent connections, allowing them to model sequences. Long Short Term Memory (LSTM) networks (Hochreiter and Schmidhuber 1997) are a kind of RNN with differentiable gates, allowing them to overcome the so-called *exploding and vanishing gradients* problem and preserve information from many time-steps in the past.

Now with increased compute power and large training sets, LSTMs and variants are having lots of success not only in sequence *classification* (Greff et al. 2015), but also in sequence *generation* (Eck and Schmidhuber 2002; Sutskever, Martens, and Hinton 2011; Sutskever 2013; Graves 2013; Nayebi and Vitelli 2015; Gregor et al. 2015; Wu and King 2016; Crnkovic-friis and Crnkovic-friis 2016).

However, sequence generation with RNNs is currently not an interactive process. Some recent implementations have used a turn-based approach, such as the online text editor Word Synth (Goodwin 2016), which allows a user to enter a seed for the RNN before it generates the next phrase. Though this is still not *realtime continuous* control of the output, let alone an *expressive* experience.

Method

We're using character based text models as it has been demonstrated that LSTMs are successful in generating character sequences (Graves 2013; Karpathy 2015). First we train a number of models on varied corpus of text. Then the system generates text character by character, and while it's outputting, we can gesturally steer the output towards the style of different models.

1. Training We train numerous LSTM models, each on a different corpus of text. These include the works of Shakespeare, Baudelaire, Nietzsche, Jane Austen, Donald Trump, the Dalai Lama, the King James Bible, assorted love song lyrics, Linux kernel C code, LaTeX source and more. Each corpus is picked due to its nature of being easily recognizable in style and content.

2. Prediction, visualization and interaction Once trained, the system generates text by: running each of the models, mixing their output probability distributions $\mathbf{p_i}$ via model mixture weights $\mathbf{w_m}$, sampling a character from the mixed probability distribution $\mathbf{p_{mix}}$, and feeding the sampled character back in as an input.

The system generates and outputs characters at roughly 10-20 chars/second. While it's outputting, we can steer the output towards different models by interacting with the system and dynamically shifting the mixture weights $\mathbf{w_m}$ for each model. Interaction is through moving the mouse cursor, moving our hand (tracked by a LeapMotion), or playing with midi sliders.

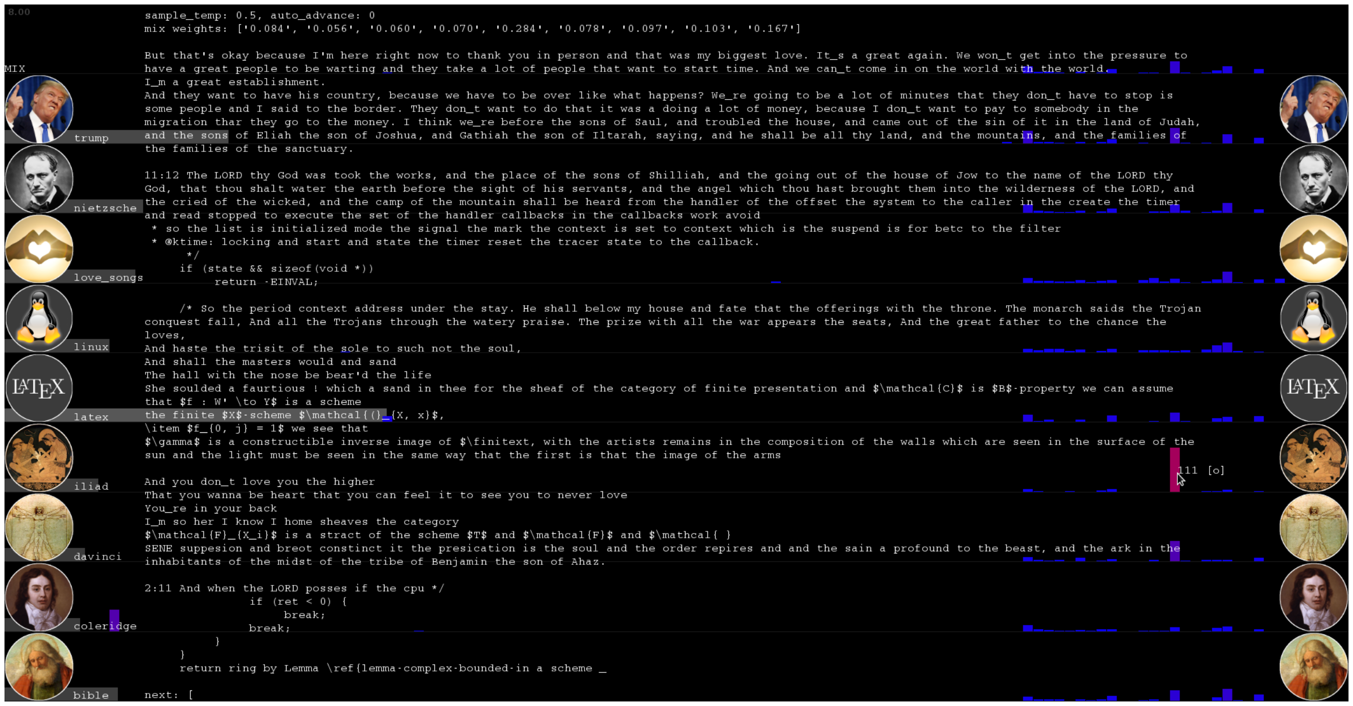
We are thus able to guide the system to morph between the different models' output with relatively smooth transitions.

Results and discussion

In this study we trained multiple models on different corpora and mixed their predicted $\mathbf{p_i}$ via $\mathbf{w_m}$, interactively controlled at every step by a user's gestures. This allows a user to continuously control the output of an LSTM as it generates a sequence, and seamlessly morph it between styles while it's generating. Figure 1 shows an example output.

The system works well and demonstrates interesting behaviour. When multiple models are active, containing words or phrases that are common to all models, the probabilities for the common next characters accumulate whilst probabilities specific to individual models are suppressed, i.e. when multiple models are active the system tends towards common words and phrases.

Sometimes, while a sequence is being generated, a particular model might have a spiking $\mathbf{p_i}$ (very high confidence for a particular character). If at that point other models have wider $\mathbf{p_i}$ (lower confidence aross multiple characters), then the first model will overpower and dominate the se-



```
sample_temp: 0.5, auto_advance: 0
mix weights: ['0.084', '0.056', '0.060', '0.070', '0.284', '0.078', '0.097', '0.103', '0.167']

But that's okay because I'm here right now to thank you in person and that was my biggest love. It_s a great again. We won_t get into the pressure to
have a great people to be warting and they take a lot of people that want to start time. And we can_t come in on the world with the world.
I_m a great establishment.
And they want to have his country, because we have to be over like what happens? We_re going to be a lot of minutes that they don_t have to stop is
some people and I said to the border. They don_t want to do that it was a doing a lot of money, because I don_t want to pay to somebody in the
migration thar they go to the money. I think we_re before the sons of Saul, and troubled the house, and came out of the sin of it in the land of Judah,
and the sons of Eliah the son of Joshua, and Gathiah the son of Iltarah, saying, and he shall be all thy land, and the mountains, and the families of
the families of the sanctuary.

11:12 The LORD thy God was took the works, and the place of the sons of Shilliah, and the going out of the house of Jow to the name of the LORD thy
God, that thou shalt water the earth before the sight of his servants, and the angel which thou hast brought them into the wilderness of the LORD, and
the cried of the wicked, and the camp of the mountain shall be heard from the handler of the offset the system to the caller in the create the timer
and read stopped to execute the set of the handler callbacks in the callbacks work avoid
 * so the list is initialized mode the signal the mark the context is set to context which is the suspend is for betc to the filter
 * @ktime: locking and start and state the timer reset the tracer state to the callback.
     */
    if (state && sizeof(void *))
        return -EINVAL;

    /* So the period context address under the stay. He shall below my house and fate that the offerings with the throne. The monarch saids the Trojan
conquest fall, And all the Trojans through the watery praise. The prize with all the war appears the seats, And the great father to the chance the
loves,
And haste the trisit of the sole to such not the soul,
And shall the masters would and sand
The hall with the nose be bear'd the life
She soulded a faurtious ! which a sand in thee for the sheaf of the category of finite presentation and $\mathcal{C}$ is $B$-property we can assume
that $f : W' \to Y$ is a scheme
the finite $X$-scheme $\mathcal{()_{X, x}$,
\item $f_{0, j} = 1$ we see that
$\gamma$ is a constructible inverse image of $\finitext, with the artists remains in the composition of the walls which are seen in the surface of the
sun and the light must be seen in the same way that the first is that the image of the arms

And you don_t love you the higher
That you wanna be heart that you can feel it to see you to never love
You_re in your back
I_m so her I know I home sheaves the category
$\mathcal{F}_{X_i}$ is a stract of the scheme $T$ and $\mathcal{F}$ and $\mathcal{ }
SENE suppesion and breot constinct it the presication is the soul and the order repires and and the sain a profound to the beast, and the ark in the
inhabitants of the midst of the tribe of Benjamin the son of Ahaz.

2:11 And when the LORD posses if the cpu */
              if (ret < 0) {
                  break;
              break;
          }
    }
    return ring by Lemma \ref{lemma-complex-bounded-in a scheme _

next: [
```

Figure 1: An example of the screen and output. Each model's output probability distributions $\mathbf{p_i}$ can be seen, as well as the model mixture weights $\mathbf{w_m}$ (horizontal grey bars) and final mixed probability distribution $\mathbf{p_{mix}}$ (top row).

quence generation. This is quite likely to start a positive feedback loop and that model will stay in control of the sequence generation until it reaches a point where its probability distribution widens, and another model spikes. So it's very possible to see hints of love songs, philosophy or poetry within C comments and variable names or LaTeX equations. It seems there are '*hand-over*' words or sequences which are common to many models, but have stronger connotations in some models over others. This is of course further guided by the user's actions, who can choose to push further towards the emerging theme, or pull towards another style and seamlessly go from one style to another over these hand-over words.

As opposed to training many models independently on different corpora, another approach we are looking at is using a single model trained on the entire corpora. Then controlling output via manipulating the internal state of the LSTM. This has advantages and disadvantages, particularly when it comes to adding a new corpus (style) to the system.

References

Crnkovic-friis, L., and Crnkovic-friis, L. 2016. Generative Choreography using Deep Learning. *arXiv preprint arXiv:1605.06921*.

Eck, D., and Schmidhuber, J. 2002. A First Look at Music Composition using LSTM Recurrent Neural Networks. *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale* 103.

Goodwin, R. 2016. Word Synth.

Graves, A. 2013. Generating sequences with Recurrent Neural Networks. *arXiv preprint arXiv:1308.0850*.

Greff, K.; Srivastava, R. K.; Koutník, J.; Steunebrink, B. R.; and Schmidhuber, J. 2015. LSTM: A Search Space Odyssey. *arXiv preprint arXiv:1503.04069*.

Gregor, K.; Danihelka, I.; Graves, A.; and Wierstra, D. 2015. DRAW: A Recurrent Neural Network For Image Generation. *arXiv preprint arXiv:1502.04623*.

Hochreiter, S., and Schmidhuber, J. 1997. Long Short-Term Memory. *Neural Computation* 9(8):1735–1780.

Karpathy, A. 2015. The Unreasonable Effectiveness of Recurrent Neural Networks.

Nayebi, A., and Vitelli, M. 2015. GRUV : Algorithmic Music Generation using Recurrent Neural Networks.

Sutskever, I.; Martens, J.; and Hinton, G. 2011. Generating Text with Recurrent Neural Networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, volume 131. 1017–1024.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems*, 3104–3112.

Sutskever, I. 2013. *Training Recurrent Neural Networks*. Ph.D. Dissertation, University of Toronto.

Wu, Z., and King, S. 2016. Investigating Gated Recurrent Neural Networks for Speech Synthesis. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5140–5144. IEEE.