Exploring Looping Effects in RNN-based Architectures

Andrei Shcherbakov[μ] Saliha Muradoğlu[ΩΦ] Ekaterina Vylomova[μ]
sandreas@unimelb.edu.au, saliha.muradgolu@anu.edu.au, ekaterina.vylomova@unimelb.edu.au
[μ] The University of Melbourne [Ω]The Australian National University (ANU) [Φ]ARC Centre of Excellence for the Dynamics of Language (CoEDL)

Morphological reinflection task

Morphological inflection is the task of generating a target word form (e.g., "runs") from its lemma ("to run") and a set of target morphosyntactic features (tags, "Verb;Present Tense;Singular;3rd Person").

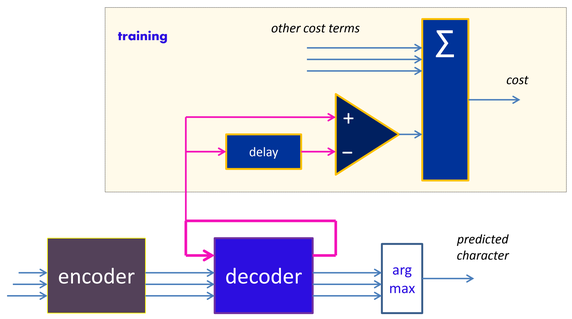
Data

Nen is a Papuan language of the Morehead-Maro (or Yam) family, spoken in the Western province of Papua New Guinea by approximately 400 people. The language is highly under-resourced, and Muradoglu et al. (2020) is the only computational work on it we are aware of, and in current study we use the data derived from their corpus.

Russian, a Slavic language from Indo-European family, on the other hand, is considered as high-resource. We use the splits from the SIGMORPHON–CoNLL 2017 shared task on morphological reinflection Cotterell et al. (2017).

	Nen	Russian
Training samples	1589	1000
Development samples	227	1000

Dataset sizes



Problem

repetitive loops, a common problem in contemporary text generation (such as machine translation, language modelling, morphological inflection) systems.

Example (Nen language reinfletion):
ynawemaylmyylmyylmyylmy-
ylmyylmyymayamawemyymamyamawemyymamya-
mawemyymamyamawemyymamyamawemyymamyam-
awemyylmyamyamawemyymamyamawemyymamya-
mawemyylmyylmyy, where the correct form is ysnewem.

Architecture

We reused the hard attention model specifically designed for the morphological reinflection task Aharoni and Goldberg (2017). The model consists of two modules; (1) an array of LSTM Hochreiter and Schmidhuber (1997) encoders and (2) an LSTM decoder.

We introduced an extra decoder output that is trained to always be increasing while new output characters are produced. More specifically, we added an extra output r and an extra input $\tilde{r}$ to the decoder. To ensure that r increases gradually while target word characters are generated, we modified calculation of total loss in the model training, allowing an extra (hinge-like) term as follows:

$$L = \max(0, \gamma \cdot (s - \Delta r)) \quad (1)$$

Here Δr is the difference between current and previous r values. Initially, for every predicted word form r is set to zero. Having observed the dynamics of r value in preliminary training experiments, we chose $\gamma = 50$; $s = 0.05$.

Options

For better exploration of different factors, we tested combinations of the following setting variations:

- ☑ feeding r back to $\tilde{r}$ vs. ☒ leaving it unused (letting $\tilde{r} = 0$)
- ☑ requiring r to increase vs. ☒ leaving it free
- ☑ scalar r vs. ? vector r
- ☒ using an external auto-incremented value for r vs. ☑ r is an extra decoder output

Summary of modes used in experiments

denotation	goal for r	$\tilde{r}$ value
n ("*none*")	none	zero
i ("*increment*")	r is ablated	incrementing
f ("*feedback*")	none	previous r
u ("*unused*")	increase	zero
s ("*all set*")	increase	previous r

Note: if r is a vector, its size is added before a mode symbol: '3s'.

Hypothesis

We hypothesized that the looping is primarily caused by merging of decoder states relevant to different word positions. Therefore, introduction of variables that are guaranteed to be different at distinct stages of output word form production should reduce looped prediction rate.

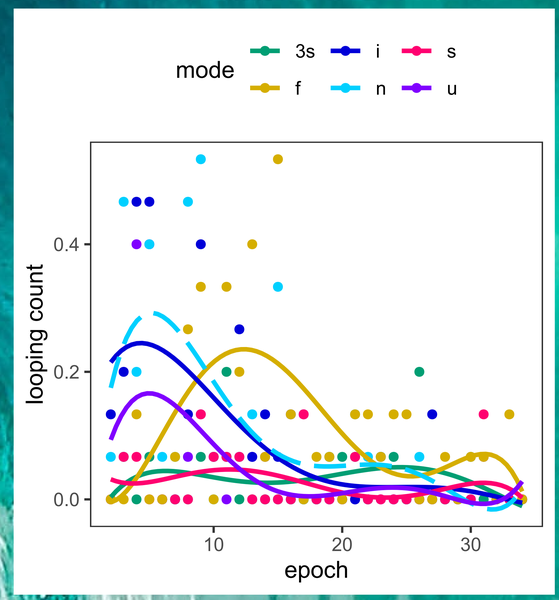
Finding

Presence of a decoder output which is trained to progressively increment reduces the average rate of looping sequences in multiple times. In most cases the positive effect is more significant if this output is fed back to the decoder.

Recommendation

(1) add an extra scalar output to the decoder
(2) endorse it to increase by inclusion a respective term into a training loss formula and
(3) feed it back as an encoder input.

Results – Nen



Results – Russian