TF-LM: TensorFlow-based Language Modeling Toolkit

Lyan Verwimp, Hugo Van hamme, Patrick Wambacq

ESAT – PSI, KU Leuven, Belgium

{lyan.verwimp, hugo.vanhamme, patrick.wambacq}@esat.kuleuven.be

1. Motivation

- ❏ Long Short-Term Memory Language Models = state of the art
  - ☹ Implementing and training neural networks from scratch = difficult and time-consuming
  - ☺ Many deep learning frameworks exist (TensorFlow, PyTorch, Keras, Caffe. . .) that usually offer the building blocks and example scripts
    - ☹ . . . but the example scripts are usually simple and not easy to extend
  - ☺ Many researchers publish their code
    - ☹ . . . but usually only publish the code for their model without offering a more general framework

⇒ We present a framework for training and testing language models

- ❏ based on the most popular deep learning framework **TensorFlow**
- ❏ with a focus on **modularity** such that new ideas can easily be tested
- ❏ providing **different training/testing** configurations
- ❏ **not** optimized for efficiency

⇒ Pre-trained LMs

`https://github.com/lverwimp/tf-lm`

2. Functionality of the toolkit

Command line switches

- ❏ Device: CPU or GPU
- ❏ Training/validation/testing

Configuration file

- ❏ Reading data: all at once or line by line (large datasets)
- ❏ Input unit:

word	*cat*
characters	*c a t*
word + characters [1]	*cat c a t*
character n-grams	*<bow>c ca at t<eow>*

- ❏ Batching:

discourse level	*owned by <unk> & <unk> co. was under contract with <unk> to make the cigarette filters <eos> the finding probably*
sentence level	<bos> the plant which is owned by <unk> & <unk> co. was under contract with <unk> to make the cigarette filters <eos> @ @ @ . . .

- ❏ Open vocabulary
- ❏ Determine vocabulary based on training text or read from file
- ❏ Graph parameters: number of layers, size of layers, batch size, number of steps to unroll for backpropagation through time
- ❏ Initialization scale (uniform distribution)
- ❏ Training:
  - Several optimizers – easy to add a new one
  - Regularization:
    - By default dropout on input embeddings and output of LSTM cell
    - By default clip norm of gradients
  - Early stopping or not
  - Exponentially decaying learning rate or not
- ❏ Unidirectional (1) or bidirectional (2):

$$y_{t+1} = g(W_o\, h_t + b_o) \tag{1}$$

$$y_{t+1} = g([W_o^f\, h_t^f\; W_o^b\, h_{t+2}^b] + b_o) \tag{2}$$

- ❏ Testing:
  - Perplexity
  - (Re-)scoring hypotheses: log probabilities
  - Predicting words given a seed word/sentence:
    - Most likely word
    - Sample from multinomial distribution of softmax
  - Generate 'debugging file' similar to SRILM [2]: can be used to calculate interpolation weights (with SRILM's *compute-best-mix*)

2. Pre-trained LMs

`http://homes.esat.kuleuven.be/~lverwimp/lstm_lm/`

- ❏ 2 English benchmarks:
  - Penn TreeBank (900k, 70k, 80k, vocab 10k)
  - WikiText (2M, 210k, 240k, vocab 33k)
- ❏ 2 corpora of spoken Dutch:
  - Corpus of Spoken Dutch (8M, 200k, 240k, vocab 100k)
  - Subtitles (Sub) dataset (45M, 100k, 120k, vocab 100k)

4. Experimental results

Perplexity

Data	5-gram	sentence	discourse
PTB	147.9	102.4	**84.1**
Wiki	231.0	150.6	**98.2**
CGN	395.2	257.6	**192.6**
Sub	114.5	74.4	**65.1**

Test perplexity of the pre-trained sentence-level and discourse-level LSTMs, compared with 5-gram LMs.

Input/output unit	PPL
word	84.1
character 2-gram	101.1
word+characters	**83.6**
character	2.8

Test perplexities for models with different input/output units on PTB.

Hypothesis rescoring

Hypothesis	Log prob
he made a sales goal he says	-31.9997
he made a sales call he says	-32.0053
he made its sales call he says	-32.0255
he made as sales call he says	-32.0360
he made it sails call he says	-32.0619
he may to a sales goal he says	-35.9857
he made a sales goal he says it	-35.9978
he made a sales call he says it	-36.0043
he made a sale to call he says	-36.0083

Rescoring 100-best lists from he DARPA WSJ'92 and WSJ'93 data sets with the pre-trained PTB LM.

! Note: word insertion penalty needed.

Predicting next word(s)

seed	**consumers may. . .**
PTB-p	be able to <unk> the <unk> of the <unk>
PTB-s	no longer be active
Wiki-p	be used to be a <unk>
Wiki-s	have made 0 - 3 victory in Mahwah
seed	**consumers may want. . .**
PTB-p	to be <unk>
PTB-s	to keep the gop golden share
Wiki-p	to be the first to be <unk>
Wiki-s	to have a strong impact on the storytelling
seed	**in recent. . .**
PTB-p	years
PTB-s	months after four years of investments last month for the year alone
Wiki-p	years
Wiki-s	years , broadcasts by Conservative Party theatre in the 18th century
seed	**The city 's growth has reflected the push and pull of many social. . .**
PTB-p	security benefits
PTB-s	states in the areas of southeast asia
Wiki-p	contexts
Wiki-s	forms

Predicting the most probable words ('-p') or sampling based on the multinomial distribution ('-s') with the pre-trained PTB and Wiki models.

Interpolation weights

Model	Valid PPL	Test PPL
5-gram	155.12	147.9
LSTM	107.5	102.4
interpolation	**98.6**	**94.7**

Optimal weights: 0.24 for n-gram – 0.76 for LSTM (pre-trained PTB model).

5. Conclusion

- ❏ Open-source toolkit for language modeling based on TensorFlow
- ❏ Focus on modularity/easy to adapt
- ❏ Several options for input/output unit, batching, training and testing
- ❏ Pre-trained LSTM LMs on English benchmarks and corpora of spoken Dutch

6. References

[1] Verwimp, L., Pelemans, J., Van hamme, H., and Wambacq, P. (2017b). Character-Word LSTM Language Models. In *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 417–427.

[2] Stolcke, A. (2002). SRILM an extensible language modeling toolkit. In *Proceedings International Conference Spoken Language Processing*, pages 901–904.

This research is funded by the Flemish government agency IWT (project 130041, SCATE).