Next Word Predictor

Simon Döbele & Thomas Bouquet

DD2417 Mini-Project Report

20.05.2022

Intro

Observations:

- One study found that smartphone typing speed is on average 36 WPM (words per minute) [1]
- Note: there is a lot of variability, e.g. between age groups.
- Still, this is on average about 15 WPM slower than typing on a computer keyboard.

Possible solution: Word prediction as a way to accelerate the user's writing.

- n-gram language model(s)
- RNN-based language model

Outline

Task

The task of next word prediction

...is to predict the next word w_t given a sequence of previous words $w_{t-n-1:t-1}$, so we want to find a probability distribution over next words given those previous words:

$$P(w_t|w_{t-n-1:t-1})$$

Task

The task of next word prediction

...is to predict the next word w_t given a sequence of previous words $w_{t-n-1:t-1}$, so we want to find a probability distribution over next words given those previous words:

$$P(w_t|w_{t-n-1:t-1})$$

Methodology:

- Using two different types of language models.
  - n-gram Model
  - Recurrent Neural Networks (RNN):
- Building a web interface with Dash

Background

The task of next word prediction

...is to predict the next word w_t given a sequence of previous words, so:

$$P(w_t|w_{t-n-1:t-1}) = \prod_t P(w_t|w_{t-n-1:t-1})$$

...uses language models

Background

The task of next word prediction

...is to predict the next word w_t given a sequence of previous words, so:

$$P(w_t|w_{t-n-1:t-1}) = \prod_t P(w_t|w_{t-n-1:t-1})$$

...uses language models

Methodology:

- Using two different NLP technologies
  - n-gram Model
    - uses the previous $n-1$ words (instead of going from 1 to $t-1$)
  - Recurrent Neural Networks (RNN):
    - uses the whole sequence of previous words (from 1 to $t-1$, i.e. $t-n-1=1$)
- Building a web interface with Dash

Background

Example

- Sentence: "What are you going to do..."
- bigram (2-gram) model: only takes into account the previous word, i.e. "do"
- RNN-based language model can take arbitrary sequence length into account

Background

N-gram

- are calculated using the relative frequency count

Background

N-gram

- are calculated using the relative frequency count

RNN

- a vanilla RNN is a Neural Network with cycles well suited to the time-based nature of language.
- we use pretrained word embeddings

Implementation

N-gram

- relative frequency count
- stupid backoff [2]: combining n-grams all the way until 1-grams
- solves the data sparsity problem (e.g. when 5-gram is not present)

Implementation

$$\mathbf{e_t} = \mathbf{E}\mathbf{x_t}$$

$$\mathbf{h_t} = \textit{tanh}(\mathbf{U}\mathbf{h_{t-1}} + \mathbf{W}\mathbf{e_t})$$

$$\mathbf{y_t} = \textit{softmax}(\mathbf{V}\mathbf{h_{t-1}})$$

RNN

- initialize each word by pretrained GloVe embeddings
- train RNN in mini-batches: passing multiple sequences of words (converted to embeddings) through a vanilla RNN
- use a final linear layer to map to the vocab size
- output probabilities for potential next words via softmax
- code reuse: adapt boilerplate code from assignment 4 to fit next word prediction

Data

What is it? Where does it come from?

- 50-dimensional pretrained **glove word embeddings**[1]
- training and testing datasets are subsets of the **News Crawl 2010 dataset**[2]
- **same training and test datasets** for both *n*-gram and RNN models for a **fair comparison**

[1] Accesible at `http://nlp.stanford.edu/data/glove.6B.zip`

[2] released as a part of ACL 2014 Ninth Workshop on Statistical Machine Translation

Demo

- Live demonstration is this way → `http://127.0.0.1:8050`

Results

Evaluation method

- Goal of application: Reduce the number of keystrokes
- Evaluation method: accuracy
- Saved keystrokes: Given previous words (and possibly an already typed number of characters), produce a fixed number W of suggestions to the user, and determine: is the correct next word part of those predictions or not?

Results

Evaluation method

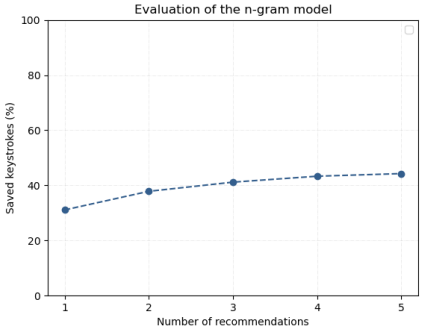
- Goal of application: Reduce the number of keystrokes
- Evaluation method: accuracy
- Saved keystrokes: Given previous words (and possibly an already typed number of characters), produce a fixed number W of suggestions to the user, and determine: is the correct next word part of those predictions or not?

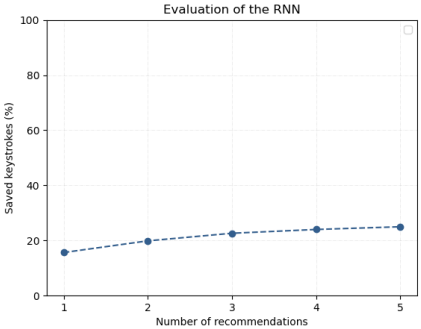
Saved keystrokes measurement

1. Process a **test set** different from the training one
2. Go through the text and use words as context / target
3. Compute how many **typed characters** are required to correctly guess the next word
4. Repeat for **different numbers** of suggested words

Results

Numerical results 1

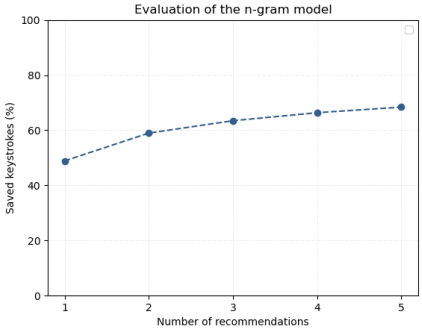


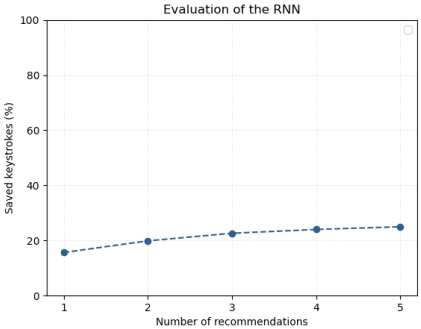


- Accuracy increases with the number of recommendations
- Better results with the ***n***-gram model (on same size dataset)

Results

Numerical results 2





- Accuracy increases with the number of recommendations
- Better results with the ***n***-gram model (on an even bigger dataset)

Discussion

- RNN takes a very long time to train to achieve good results
- N-gram ”overfits” on training data, we should expect bad generalization
- RNN expected to generalize better to different datasets
- Note: for our use case: predicting next words for a smartphone user, choose n-gram model, because ”overfitting on the user data” not a bug but a feature!

Conclusions

- Satisfying results with **good accuracy** in prediction
- *n*-gram model perform better; **more suited to the smartphone usage context**
  - less memory storage, easy access, ...
- Future research: compare generalization capability on different datasets
- optimize RNN performance via hyperparameter tuning and training for longer periods of time and on more data

THANK YOU!

References

[1] Kseniia Palin, Anna Feit, Sunjun Kim, Per Ola Kristensson, and Antti Oulasvirta. How do People Type on Mobile Devices? Observations from a Study with 37,000 Volunteers. In *Proceedings of 21st International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI'19)*, 2019.

[2] Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 858–867, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

Please find further references in our corresponding article.