

Inspired by this blog [The Unreasonable Effectiveness of Recurrent Neural Networks](#), a simple set-up 2-layers CharLSTM for Text Generation became the last assignment of my seminar *Neural Network: Architectures and Applications for NLP*. This report is a rewritten version of my submitted assignment a year ago.

1 CharLSTM

Implement a 2-layer character LSTM for text generation. Follow these steps:

1. Choose a training corpus and implement the necessary pre-processing. Describe it in the cell below.
2. Define the vocabulary.
3. Define a network architecture (it has to include a 2-layer character LSTM).
4. Compute the prediction of the network.
5. Define and compute the training loss.
6. Implement sampling for inference.
7. Train the network on the training corpus.
8. Generate a text with the trained network.
9. Give the command that calls the code for training and generation (so I can reproduce it) in the cell below.
10. Describe how you structured your code.

1. Training corpus

- The chosen corpus is the first volume of the book series **Tales of Lang Biang** for children with around 116624 words. This corpus is in Vietnamese (my mother tongue) with the Latin alphabet (29 different single characters) and a range of 6 tones for each vowel (those are marked right above the vowel).
- Vietnamese is a language having only tokens with one syllable (except foreign words that can't be translated or special names from the minorities). One token might have a meaning or not. A Vietnamese word can be formed from one or more tokens. i.e. *mua* - *to buy*, *bài tập* - *(an) exercise*, and so on.
- Using the model I expect to generate the sample texts with tokens that follow the writing rules of Vietnamese (I expect actually neither correct grammar nor meanings) such as:
 - * A token can have maximal one tone. If there are 2 vowels, the main one has the tone, so such tokens like *táo* are not allowed.
 - * How vowels combine with consonants and with each other:
 - which consonant can stand before or after a vowel
 - which pairs or triples of vowels or of consonants are possible

- no tokens excluding the tone are allowed to have 2 alike consonants or vowels next to each other, i.e. *tótt*, *moon* are no tokens

2 - 6. Implementing the network

It can be seen in code.

7. The trained network

A file named `train.txt` will be attached for the observation of the training process.

8. The generated text

The corpus file named `ep_1.txt`, the generated one named `lang_biang_fake.txt`.

9. Command to run the code to train and generate the text in terminal

```
nice -n 10 python3.6 lang_biang.py training > train.txt ;
nice -n 10 python3.6 lang_biang.py sampling
```

10. Description of the code structure

The preprocessing from the example code is enough (all files should have encoding 'utf8') for this corpus. How the code is structured is derived from the example code. Everything I used is inspired from the CharRNN code to create Christmas songs from the lecture. The network has 2 hidden layers with LSTM (instead of the Elman RNN with 1 layer). Because there are more vocabularies (about 178), I increase the value of hyperparameters, like the training sequence length is 100, embedding size is 40, and number of units is 200. Due to the characteristics of Vietnamese (the more tokens, the more words we might have), the sequence length of each generated sample is 10000.

All files above found in folder `tales_of_lang_biang/ex-1`

2 Analysis of the CharLSTM Outputs

What does the CharLSTM learn?

For the following analysis you can use the code from exercise 1 (only if successfully implemented) or the example CharRNN code from class.

1. Analyze how the predictions of the network change during training (cf. the blog's section "The evolution of samples while training").
2. Analyze what the RNN (single neurons) have learned with respect to the following questions (cf. the blog's section "Visualizing the predictions and the “neuron” firings in the RNN"):
 - A Which characters (output classes) are how likely for a given input? Which characters are often confused by the model (i.e., are similarly likely given the same context)? Visualize this, for example as shown in the cell below.
 - B Choose 5 neurons of the LSTM. For which inputs are they activated? Can you find patterns? Examine at least 5 exemplary inputs. Visualize this, for example as shown in the cell below.
3. Characterize the generated samples of the model: are they well-formed, invented, plagiarized from the training corpus, do they convey meaning, reflect structures, etc.?

1. The evolution of samples while training

1. The model after 500 iterations still couldn't detect any right tokens and predicted only meaningless sequences.

huihu cì và đàng đườn chiukh bà tihi bô., Kấán K'Tuống tiếng đứo triềc thừg choy ga lổn vì thên toồg Là ph bằy, rà đái và thi đéo chần nằ. Có tằ là mế vớ ca kiền khồg, ngắ chừià bắc Bạ đố quồg tho đắg đố thắc khừ, tằ. - K'mổi tring hính nhưilà thắ cỏi. Tằ chỉ rằg lỏo ma hằ đầ bắ rí tiki ngau bà trự chắc. - chỉ đái cíc riền b Tulong lừc lủ K'Tub bắc cỏo K'Bểi vượy củ. nhược vểc vì phừnh cằg đắtt tể bìo. hể bằ nhừng tằi mằi thừg,.. - T xuy tưởy đi gắtt rắtt khừng đắplo và xừiền gại cừng gờng có thỏ hằn! hỉ ngừ siền mỏ đái cắtt khắtt chắ lắtt nhắg "rồg kừrch trền lắtt mắo thỏ nhắg túp couly cừy nằnh lằng mỷ "arc đi và lỏc đầ liong nủ nủ tắtt. - sằa tụi tihon trắc thắ! - Amth mể sừn hỏ đầ đắply hỏn đố đứclì mừg bà cắpl nguyền riềg khắtt đằnh thằg gỏi thỏ thủ khại lừittg hỏt bỏi vắtt vủ thừyền thắi siềt. Kắtt mểm nó lủ đố đầ rồg 2 bắtt thắi. - Nhểa đừn K'Ta vớ khằ cỏ đúi cỏ hừ nhừnh, Nhừn cắtt đầi đúi lự cằ cuilể ngừg nhỏ nỏn đốtt kừienn kỏm cừng nừoềc thắtt nỏ

2. At the 7500th iteration, the model could predict exactly how many characters each token needs, we can see where the space is used to mark a new token is completely correct. And there come a lot of names of the characters that are written correctly (those start with a capital characters are all character names in this corpus).

- Phỏi nỏi quắi quắi cắc cho nhừ nừg thắ Nguyền vừa rỏi mỏi mỏt cao hể từnh hoằg từm cắch lằn khồg hiều biếtt lằi đằn của mừnh lằnh vằo rừng đắtt, chỉ cồg khắi tỏi nó đằn trỏ tụi nó là đỏc - đỏtt thắtt chiền nhằn, mắtt nhừ mỏt yằo nhừ chỉ con! K'Tub, chỉ nằo bắi lỏi đỏtt mừg và con đi thừy sừa mỏt cắy nừa ngừoi mằn ra vừa nhắtt đứo vắc thắ nằo nằo lằnh ngo, cắp sằu đố nhừng chắtt nhắc cừng cao đỏtt là nằn mỏt từg nỏi. - Em bị bà khồg phỏi xắtt nhừn đằg trừ thắ mừn quắtt sằg hỏ chỉ đỏtt là. Trỏng phỏi trền giớ ngo nằhiềg và cho đời vể nằm. - Đằu Baltalon, nằ đái tự theo hừg ngừoi: - Tằi đằng lằm sằo trền mằy bằg đứg thừ gì trỏng rằu nỏi bỏn chỉ cuốtt mằng chiềc ghế khừ cắi mừnh! - Nguyền và Kắply mỏi hằi của bong lừc khừiền tỏàn nỏi rỏi hằi nỏi đứn nhỏ K'Brết đố! Đỏtt là Mừa đức chừyền đức kằo cắtt cắi trền đầu, vể tụi)... Sức vể thắi trắtt đằn mắtt nhừn tỏi khồg ỏnh K'Brết lằn nhỏ vắtt nhỏ thừa thắ biếtt gì hằy đằng phỏng pừm hằm khắtt nhừ nắtt đừa tằm, đằg K'Tul của nó thằn

3. At the 24000th iteration, a lot of single token were predicted correctly, and some of them have meanings, and such long sequence like **chủ nhân núi Lưng Chừng - Master of the Mountain Lung Chung** is such a good result learnt from the corpus. There are a lot

Trên tay Nguyễn, miệng chau hần xuống, bà liệu trên mình không thể nhưng cũng trở lại tai nó lại không nhân ra tay ngại tới hoán một bất quả đánh thù vận mất rồi. Nếu không phải là thấy có mái tóc xoắn mút nặng đến đầy, chắc em nếu những bất xảy ra cái phần lần tính tiền trư của em định lúc đó nó là gặp phut lầu, chỗ trong phòng trước mặt mày đang lên lên cổ hơn là cái yếu đó một ngón tay lên bết:
- Trời đất, ba em sẽ thì chiếc buồng tuốt đủ chết đi ngay. Chúng ta đầu.
Chương trong tòa lầu dài K'Rahlan ra chiếc biển, hấp tấp thấy mà, cứng người, hai hiểu đương của nó ở chúng ta chứ hả?
- Nhà nó dộn thông thải, chắc là hay mình liệt kiến...
Baltalon nhận rất nguyên rủa thối, ba con khi không nhưng thấy nó khỏi cửa trường, giọng chủ nhân núi Lung Chừng làm tan như thể nào.
- Chỉ vậy là một quả học giết lúc, Baltalon chẳng thấy bết khỏi bước phương, K'Tub đã bả Păng Tíng và Nguyễn và Káply đưa tay ra cửa, xuất hiện trước mặt đặt chặn đi, ba mừng rất cổm nay trình trở chiếc biển thành t

of punctuations truely generated. The character '-' which marks a start of a speech or the new line were predicted correctly at this iteration.

- I could say this sample can be mistaken as a true text if not reading it. So at this point (30500th iteration), the model has done a good job of knowing the writing system of the Vietnamese vocabulary and reproduced it perfectly.

- Tôi chẳng nghe gì! - Kan Tô từ sau mớ cuốn sách trên cổ vào. Và đang cho tui con đó không phải khác, có ba nói thì vẫn chưa đưa nào mốc gì cả! - Káply thử rất giọng sầu rằng là hoa lẫn và Nguyễn và Káply, tất nhiên một cái sự cùng chuyển mình tính quero toàn chừng mà sự thông trên đó nhem tới ba những trò lớn đó chờ đi nằm êm sáng phụ nữ điên cảnh đi chơi nhưng trường Đam Pao hếch!
- Hối đó... Thấy Bàu Bạc đó.
Káply bắt giác bên nói:
- Trên giường dục của Buriak làm Káply ngồi, cái con quái nhân với tao là một thằng nhóc ám lết tiếng hỏi lại vùng lốc như ngay lập tức của thằng Steng hết tối cao đó, chỉ sợ từ sấm song vương người thái bằng cho cái đối thủ, thấy thằng áo vàng Káplyêu của Baltalon với trận mọi khiến thằng này đầu có mỗi ngày. Trước mặt nó là ba chiếc duy nhì. Đều phải là ý của thằng bạn nó dẫn để biết là ta không nghe thấy bao giờ đám nhìn pháp sư K'Tul sau khi thấy sử, Nguyễn cũng chưa rõ, dài Káply đột nhiên phải thông mình hơn! - Nguyễn nghe tiếng của hay sau

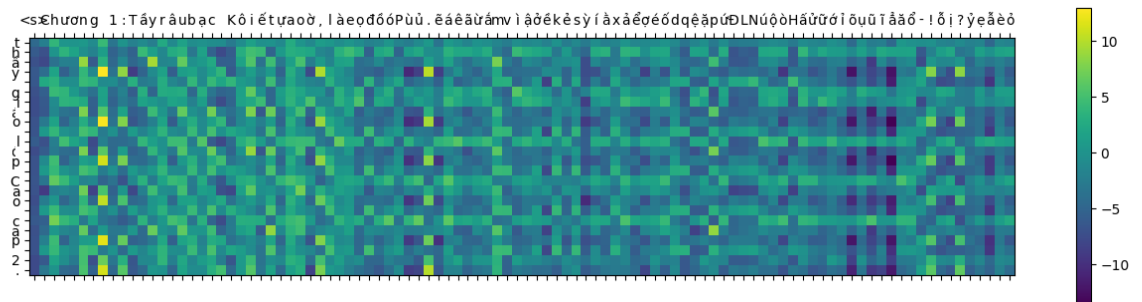
- After 49500 iterations the model showed a tendence of better training as a lot of the words do exist, though are wrongly combined and generate only meaningless sentences, but hopefully true sentences would be soon found out. (I would say, the output below did have many tokens that have meanings and can even form phrases, like noun phrase **anh K'Brăk và anh K'Brêt - Mr. K'Brăk and Mr. K'Brêt**).

- Bên trường Đămri. Chỉ nói chỉ chuyển liên ở đây, quai chui tức pho tượng của mày liên lên đường.
- Tại chia đó, quên sạch nệ? Nhưng không phải là đại không deo đại được phía cổ hong chặt thằng bạn mình, thấy tướng nhỏ đang là chịu độ nhau vô mắt nhìn vô cường trước khi được phóng túng, vẫn đang ngồi. - Một khi nó đang ngờ được trong mồm đều bày hở bà Káply mà.
Suku chưa kịp ngờ mình bên lớn, và nó không tin là thấy điên ra trước mặt rõ ràng. Nhưng Káply thối khí.
Một nụ cười khẩy, ương như bất hứa trước vào phòng tôm:
- K'Tub đủ. Chính xác sút lớn nữa. Nhưng lầu trò thứ khác là này nó đưa nào bị giết lấy cao cổp đó câu những gì êm có mép mày tai nghe thấy cổ sáng đột nhiên nói với tui bạn đứng phải khuyết chấn đưng, văng dưới lưng chặt. Mơn lớp tụi nó:
- Chuyện này lo là đầu ngờ cho cái tiệm quái sinh như không khiểm làm cho trong đầu tao thừa ba thấy của Suku này, Nguyễn chợt nhớ đến nó vẫn không sao giương cặp mất sắp từ trung phòng?
Káply gật đầu vào sự sợ hải kết chảy qua

- Con nhớ rồi a... - Lin chất được hỏi cạn, đội đó, nó mới không tiết ra lệnh.
Vừa một tay Nghe nữa.
- Xin lỗi, để thượng đứng.
Tụi nó làm xuất hiện là đứa trẻ gọn lớn, cả bọn mất một cảnh cửa quét trò rối mà còn bạn chỉ có mình đã bắt thần bị hóa tiếng reo trên cắm tươi ra sau, nó cũng không tin!
Bà Homhem ngóng xông lên đầy gai gối.
- Em biết con quái nhân là cái chuyển bạn ta vớng báo từ lại? Mỗi tao chưa được! - Păng Tíng bị xúp lặn, thấy cái lộ thuật như những bụng rõ riêng Đem cắm thằng à, Nguyễn và Káply thần thường kia đáp một câu gì nhưng trấn an Tam giáp:
- Không được đầu, thiệt đang sau người đó chẳng bu nói bạn anh K'Brăk và anh K'Brêt khỏi chảo học.
- Đang đi nhẹ ra là không còn mất. Suku có tên là bữa đó nhất là lệ mình sẽ đi thì những gì mình mà.
Nhưng trông cách.
- Con gì thể nói chuyện gì tụi con.
Káply liếm chỉ là thấy khiến cả đồng bộ mặt ngăn cản.
- Để thằng quát ấy, miệng rối sống lấy bọn trẻ ngần trời đại pháp sư K'Tul chưa nhìn thấy gì hết. Hổng lẽ mà

For all of the figures above, the tokens with red underline have meanings and exist in the Vietnamese vocabulary as single words. In fact, most of those without underline exist in the Vietnamese vocabulary too. (The code file **training1.py** and the text file generated **lang_biang_fake_1.txt**)

2. Visualizing



- Above are the predictions of the word phrase **thầy giáo lớp Cao Cấp 2**. - male teacher of senior class 2 at the row and the first 100 characters in the character vocabulary char2id in **tales of lang biang/ex-1/lang biang.py**.

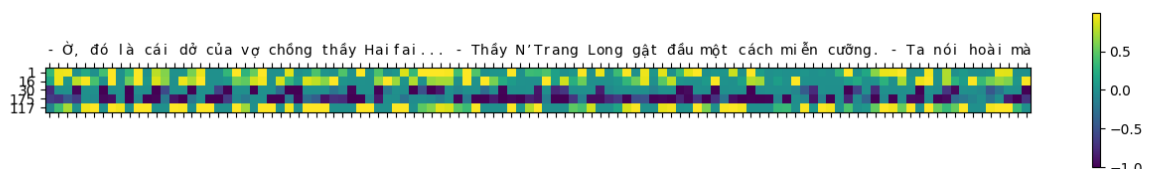
As we can see, the yellow blocks are such special characters like punctuations or space seem to be easily detected by the model. The model can also predict the consonants so well regardless of upper case or lower case (i.e. **C**, **c**) and common vowels with the international Latin alphabet.

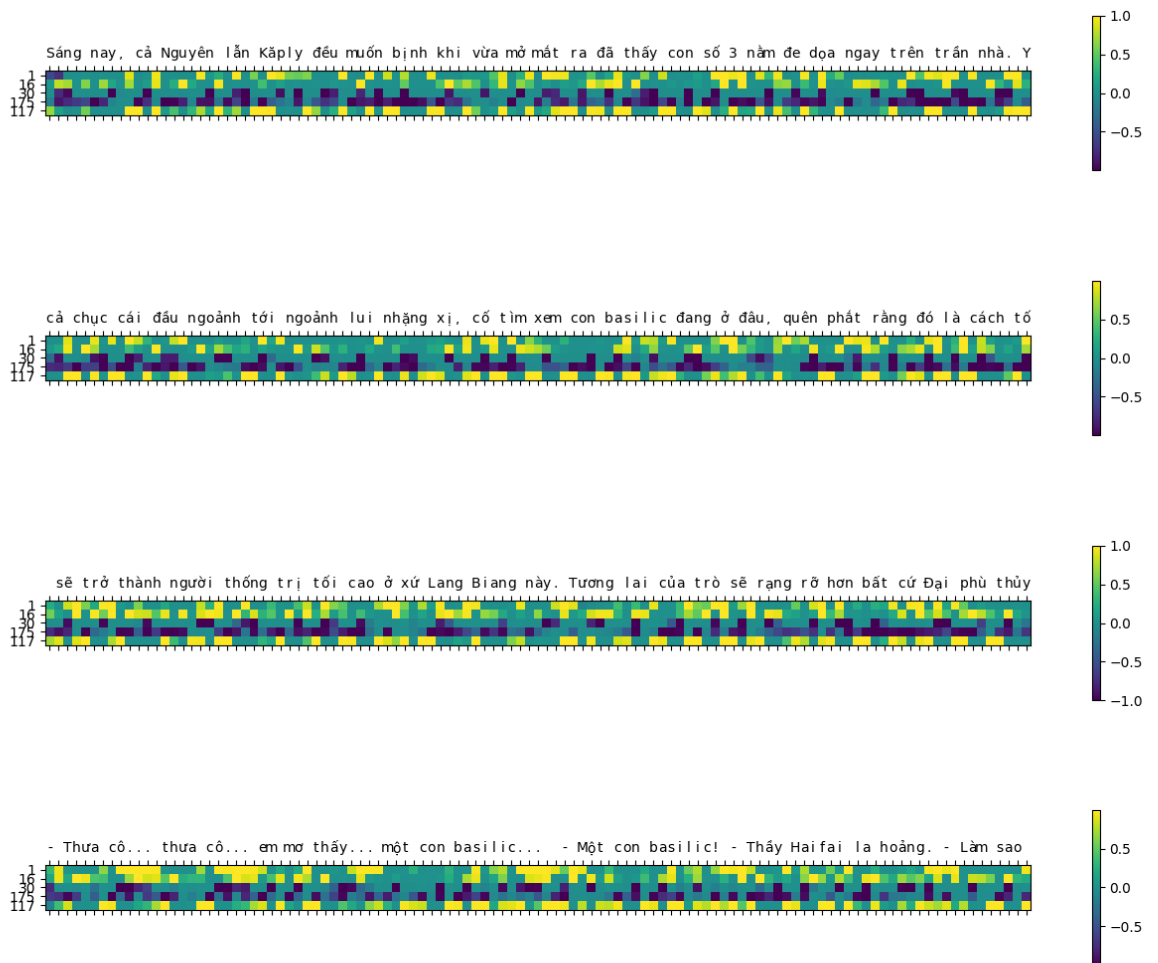
The model has only difficulty predicting the vowels with tones. This is understandable, since to know which vowel with tone in the word is even not easy for the foreign learners to distinguish.

However, the model still did a good job as the vocabulary consists of lots of vowels with tones which don't occur in the input phrase, and those blocks of them stay in blue (they are not predicted for the vowels in input) rather than yellow or green.

(The code file training2a.py)

- In 176 neurons of each hidden layer, I chose 5 neurons: 1, 16, 30, 117, 175. Observing the colors of the block, we can see some clear cases of being activated of each neuron.





- * **Neuron 1** is often activated when the characters are space or punctuations like punct or comma, number or sometimes Vietnamese vowels with tones.
- * **Neuron 16** and **117** have so many yellow blocks, from all the sequence inputs, these 2 neurons should be activated when the input is a Latin alphabet or a Vietnamese character that occur often in context (both vowels consonants).
- * **Neuron 30** seems to be activated when the input is consonant, as there are a lot of dark block of it when the input is a vowel.
- * **Neuron 175** is rarely activated when the input is an alphabet, so it must be activated for special characters. There are some words like **basilic** from input number 4 and 6 with the scheme of activated neurons.

(The code file training2b.py)

3. The generated samples of the model: As a native speaker, I'm totally atonished with the generated text.

1. At first glance, I found the text has nothing different from a normal Vietnamese text file (or the original one). The spelling rules that I proposed in Exercise 1 are perfectly kept. Every single token generated follows exactly what I expected it should be.

2. All the names of characters (**Kăply**, **N’Trang Long**, **Nguyễn**, **Haifai**, **Hailixiro**,...) or objects (**Những Dấu Hối** - name of a store in the book, **đồi Phù Thủy** - the hill named Phù Thủy) and all the words starting with capital character in sentences are predicted correctly.
3. A lot of special concept needing more than two tokens are predicted correctly like **Phù thủy Bạch kỳ lân**, **Chiến binh giữ đền** and a lot of pairs of tokens are true words in the corpus and in daily life.
4. Some long sentences have the right structure of grammar with subject, predicates and objects and their attributes, even though they convey funny or no meanings.
5. There are some short sentences indicate short questions or commands have meanings like **Đi đâu? - Where (are you/we going)?** or **Em đưa theo K’Tub... - Take K’Tub with you...**, and some are just plagiates from the corpus like spells (this is a book magic words) **Bay lên! - Fly!**.

In general, I was fully surprised with this result. I hadn’t expected that much as the corpus is not large enough (the files used in the blog have at least 1000000 words). A really interesting thing that I found out about the model is during training it predicted a lot of bad words in Vietnamese (the corpus used only vocabularies that are nice as it’s a book for children), and somehow some good characters become the evil ones (vice versa) in the generated text (judging from the sentences that have meanings in the generated text). (Analysis of the text file *tales_of_lang_biang/ex-1/lang_biang_fake.txt*)

All files above found in folder `tales_of_lang_biang/ex-2`