Progress Report

Independent Study

201401094
Satyam Pandey

To,
Professor Manish Srivastava

Table of Content

Generating factoid question in code-mixed language.

My original objective was to analysis of model trained in this paper "Generating Factoid Questions With Recurrent Neural Networks: The 30M Factoid Question-Answer Corpus" on other knowledge graph such as Reverb and Dbpedia. But later was motivated to work on problem on generating Hindi-English code-mixed factoid question. The problem is intriguing as the code-mixing has different textures based on what language is used for building the base sentence to be used as context.

My report describes my approach to the problem. It contains 3 parts, namely:

1. **Data generation [Intended part of my independent Study]**
2. **Idea with reference to previous work**

Data Generation :

I couldn't find much code-mixed data for questions online. The data provided by the social media is not that important because there are much question on these platform apart from that the language other than english are generally spelled in english leading to ambiguous spelling but here we are not concerned with spelling but more with structural semantics of the language.

So, idea for gathering the data was to heuristically manufacture a few of the code-mixed question from a proper question and then rank them with the help of humans to identify the structure of code-mixing that are generally followed in speech. This will produce a bag of code-mixed question for every question.

I observed that when we think about code-mix sentences between 2 languages there is difference in structure of sentence based on what language is used for forming the base structure. Other observation was while we substitute words between the languages there are certain parts of language that are not substituted based on the language that is used for forming the base sentence, for example in english part of speech such proper noun, auxiliary

verbs , prepositions are generally not substituted because these words form the structure of sentence. Using this I wrote codes that could generate these code-mixed sentence using tools such as Google Translate, spacy etc. My approach for generating the sentence in Hindi-English and English-Hindi codemix is described below.

Hindi-English codemix question:

In this format Hindi sentence form the base structure of the question and then some of the words in it are later replaced using the english words to generate code-mix sentence. In this I first translated simple english question into hindi using google translate and then tokenize the hindi sentence. Using google translate, translate the individual token, check if the lemma of the any token in the given english sentence have the same lemma as the lemma of individual translated token. If they match check if it's not some prohibited pos-tag etc.

The above algorithm produced, quite good code-mixed sentence as can be seen in the below example.

src: what country is the movie shanghai mystery based in
code-mix sentences:

1. फिल्म **shanghai** रहस्य किस देश में आधारित है
2. फिल्म **shanghai mystery** किस देश में आधारित है
3. फिल्म **shanghai mystery** किस **country** में आधारित है
4. फिल्म **shanghai mystery** किस **country** में **based** है
5. फिल्म **shanghai mystery** किस देश में **based** है
6. फिल्म **shanghai** रहस्य किस **country** में आधारित है
7. फिल्म **shanghai** रहस्य किस **country** में **based** है
8. फिल्म **shanghai** रहस्य किस देश में **based** है
9. फिल्म शंघाई **mystery** किस देश में आधारित है
10. फिल्म शंघाई **mystery** किस **country** में आधारित है
11. फिल्म शंघाई **mystery** किस **country** में **based** है
12. फिल्म शंघाई **mystery** किस देश में **based** है
13. फिल्म शंघाई रहस्य किस **country** में आधारित है
14. फिल्म शंघाई रहस्य किस **country** में **based** है
15. फिल्म शंघाई रहस्य किस देश में **based** है

English-Hindi Codemix sentences

In this format english is used as base language and then some of the words in this english sentence are later replaced using the hindi words to generate code-mix sentence. In this first, I first tokenised the given english sentence and then selected the token that can be replaced by

checking if they aren't any prohibited tag, pos-tag etc. Then I translated these tags and generated the sentence. The quality suffers a little as compared to Hindi-english codemix generation as there is no cross-referencing in this. But, still it gives fair enough results as shown below.

src: who was the ruler of the bahamas?
code-mix sentences:

1. कौन was the ruler of the bahamas?
2. कौन था the ruler of the bahamas?
3. कौन था ruler of the bahamas?
4. कौन था ruler of bahamas?
5. कौन था the शासक of the bahamas?
6. कौन था the शासक of bahamas?
7. कौन था the ruler of bahamas?
8. कौन was ruler of the bahamas?
9. कौन was ruler of bahamas?
10. कौन was the शासक of the bahamas?
11. कौन was the शासक of bahamas?
12. कौन was the ruler of bahamas?
13. who था the ruler of the bahamas?
14. who था ruler of the bahamas?
15. who था ruler of bahamas?
16. who था the शासक of the bahamas?
17. who था the शासक of bahamas?
18. who था the ruler of bahamas?
19. who was ruler of the bahamas?
20. who was ruler of bahamas?
21. who was the शासक of the bahamas?
22. who was the शासक of bahamas?
23. who was the ruler of bahamas?

Idea with reference to previous work

Reference: Generating Factoid Questions With Recurrent Neural Networks: The 30M Factoid Question-Answer Corpus

Their models consisted of two components: an encoder, which encodes the source phrase into one or several fixed-size vectors, and a decoder, which decodes the target phrase based on the results of the encoder.

In contrast to the neural machine translation framework, source language is not a proper language but instead a sequence of three variables making up a fact. The encoder sub-model, encodes each atom of the fact into an embedding. Each atom {s, r, o}, maystand for subject, relationship and object, respectively, of a fact F = (s, r, o) is represented as a 1-of-K vector xatom, whose embedding is obtained as $e_{atom} = E_{in}x_{atom}$, where $E_{in} \in R^{DEnc * k}$ is the embedding matrix of the input vocabulary and K is the size of that vocabulary. The encoder transforms this embedding into $Enc(F)_{atom} \in R^{HDec}$ as $Enc(F)_{atom} = W_{Enc}e_{atom}$, where $W_{Enc} \in R^{HDec * DEnc}$. This embedding matrix, E_{in}, could be another parameter of the model to be learned.

For the decoder, I'm planning to use LSTM as opposed to GRU used by them with error function E described as follows. Let *S* be the set of all code-mixed questions that are considered to be the proper sentence for the given question *q*.

$$E(Y) = min_{x \in s}(loss(Y, x))$$

where *Y* is the generated output by the decoder.

loss(Y,x) can be any loss function depending on how he distance in the space of code-mixed sentence is matricised.

An example of loss function could be L1 norm with number of words that do not match.

$$loss(Y, x) = \sum_{i=0}^{n} I(w_{Y_i}, w_{x_i})$$

where n is the maximum number of words in the sentence *Y* and *x* .

and *I(a,b)* is function like this

$$I(a, b) = 1 \; if \; a != b$$

$$else \; I(a, b) = 0$$