Progress Report (semester 2)

Template based generation of Wikipedia articles

Ankita Maity

1 Introduction

All the steps outlined in the IndicWiki manual were followed to create the Wikipedia articles on mobile devices. As such, only some external details not covered in depth in the manual and some details that were specific to my domain have been outlined below.

2 Data Collection and Visualization

Data scrapping needs to be done only after consulting privacy policies and also understanding if the information can be used in Wikipedia articles (whether they are copyrighted or not). Web etiquette needs to be followed during scrapping, like using time.sleep.

Data was used from GSM Arena. Kaggle datasets that had already done this was used with some modifications to fit the task at hand. Wikidata was also explored by using SPARQL queries. This was especially useful for collecting usable images without worrying about copyright permissions. References to Wikidata category and specific GSMArena page were collected to add to the references section later on. At the end of this process, the data collected was merged into a master dataset, and 45 attributes were kept in total.

Attributes can be explored with SweetViz. For attributes that change with time (like the current software version of a phone/number of current users), a date needs to be put in the template (like "as of May 2022"). Mostly dense attributes were kept in the final dataset, but sparse attributes can also be kept if they are important enough...if-else statements can be used later in the template to deal with the spareness.

3 Translation and Transliteration

Different libraries work better for different attributes. Neural models seem to give better results for most attributes but lead to a higher execution time. Tokenisation can be used to group similar tokens across attributes so that these need only be passed once to the library functions. They can later be grouped together to get the original attribute back. "Find and replace functionality" can also be used for this purpose. If the transliterated version is hard to read (especially for technical terms), then the original English attribute can be kept next to it in the template.

- Translation – EasyNMT("m2m_100_418M") then Indic transliteration for the digits.
- Transliteration – Indic transliteration seems to work well for most of the attributes.

Some attributes were translated or transliterated incorrectly, and Azure API was used to correct these (to be used carefully since limited credits are available in the student account). Some neural models can give incorrect values for the numerals - regular expressions can be used to correct these.

Other models were explored but were not used due to the following reasons:

- Google Translate: Anuvaad/EasyNMT gave better results for my data than Google Translate.
- Google Transliteration (google-transliteration-api): Gives good results for the digits only but works poorly for the textual content.
- DeepTranslit: Faced some version issues. It seems to work best with tensorflow 1.14 and keras 2.2.4.
- Anuvaad: This gives good results, but the run time was too long.

4 Jinja Template and Sample Article Creation

English and Hindi templates were created in the form of word documents first, which were later modified based on the feedback of the annotator and Kasyap sir. The sentences from this word document were later used to create the Jinja template. Some key details that were used to create this are given below:

- Jinja template has to be made taking into account a lot of edge cases (like missing attributes/which sentences can be randomised and which can't etc.) so that good articles can be generated from them.
- Short descriptions can be used to make the articles easier to search.
- It is preferable that the infobox template be already present on Wikipedia - this is better than creating a template manually from scratch.

- At least one category needs to be present which is not already in the Hindi Wikipedia.
- Instructions given by the Wikipedia community to IIIT Hyderabad should be followed while creating the Jinja template.

The output of the Jinja template was inspected for some rows in the dataset. Such outputs can be pasted in the Hindi Wikipedia sandbox/HiWiki sandbox to check for rendering issues if any.

5 XML Dump and Importing the Articles

The XML dump can be generated by taking the rendered text and appropriate HiWiki headers. Smaller dumps (for my domain, it was five separate XML files with about 2000 phones in each file) should be generated instead of a single big dump so that it is easier to upload. They can be imported to HiWiki after this (9519 articles were successfully imported to HiWiki for mobile phones domain).