Evaluating and Developing Methods of Generating Code-Switched Data
07-300, Fall 2021

Abhishek Vijayakumar,
under Prof. Alan W. Black, Language Technologies Institute
https://inkyubeytor.github.io/category/code-switching.html

December 7, 2021

1 Progress Report

I have not read any new papers for this project since the project proposal, as the reading of those papers allowed me to identify the set of methods I wish to evaluate during the course of my project. I have since moved on to attempting to implement these methods.

I have implemented a framework for the substitutive code-switching methods I described in my previous report. This allows me to implement substitutive methods as variations on two components: a method of selecting spans of text to substitute and a method of substituting spans from one language into another. I currently have two selection methods in place: token selection and single span selection. I also have 2 span substitution methods in place: bilingual lexicon substitution and Google Translate API substitution. These cover several of the substitutive cases I identified, such as the LEX method from the Translation for Code-Switching (TCS) paper in which individual tokens are translated [4].

I have obtained at least partial code for 3 different generative methods: Variational Autoencoders for Code-Switching (VACS), a neural method [3]; TCS, another neural method [4], and a method used for pretraining NMT (neural machine translation) [5]. Of these, the code and data for the VACS method is complete enough that I have been able to run it with minor changes. I have not yet attempted to reproduce the results from the VACS paper due to issues with compute resources detailed in the Resources section. The code for the TCS method is relatively complete, but the authors are unable to release the entire dataset. As a result, I could not run this method until I collected some real code-switching data. The code for the NMT method is incomplete, requiring the installation of dependencies that no longer exist in the indicated repositories.

I constructed a dataset by scraping the website Hinglishpedia, a blog containing Hindi-English code-switching articles in romanized form. I then used Google's transliteration API to map words identified as romanized Hindi to their Devanagari counterparts. I used this data to train a Devanagari-Latin transliteration model (an encoder-decoder model using PyTorch), which has a whole-word accuracy of 55% and unigram/bigram accuracies of over 90% and over 80% respectively.

I have also begun to implement evaluation metrics for the methods I am comparing. I have not yet been able to locally run the GLUECoS evaluation discussed in my project proposal, as the only currently available version of it requires submitting results as pull requests to a GitHub

repository for scoring [2]. However, I have been able to implement several other evaluation metrics. I have implemented the Code-Mixing Index and the BERTScore, which are both used in evaluations in several of the papers I have read [1] [6].

2 Reflection on Initial Plan

2.1 Major changes

There have been no major changes in what I plan to accomplish at this time.

2.2 Meeting your milestone

My first milestone was to implement or acquire working code for 4 of the generative methods I wish to evaluate and to construct the datasets I need.

I wrote implementations for a set of related methods that could be counted as 3 separate methods. I also now have a working version of code for one of the neural methods. I then scraped the website Hinglishpedia and turned this into a Hindi-English code-switching dataset. Given that I have completed these above steps, I believe that I have met my original milestone.

2.3 Surprises

One major surprise for me was that many code-switching generative methods actually generate text in a mix of scripts (e.g. mixing Devanagari Hindi and romanized English). As this does not reflect the way people code-switch in real-life text, I did have to take the additional step of creating a transliteration model between the scripts in order to create realistic code-switched text.

2.4 Revisions to your 07-400 milestones

As I am currently not stuck on anything and have paths forward to resolving all issues identified in earlier parts of this update, I am planning to maintain my same set of milestones.

2.5 Resources needed

I currently have access to a machine with a GTX 1070, which I will be using as compute resources for my project. The machine is currently experiencing errors with the drivers and libraries I need to work with the GPU. I am currently planning to meet with the graduate student in charge of maintaining this machine so that I can work with her to reinstall the necessary components.

I have collected the majority of the data I need for my project. There is one set of real code-switching data that I cannot obtain access to because the authors of the paper are not able to release it to others. I am currently planning to use a dataset I collected myself to replace this, but I will need more time to evaluate if it is a suitable replacement.

I have access to all other resources I will be using, such as datasets from other papers.

References

[1] Björn Gambäck. On measuring the complexity of code-mixing. 2014.

[2] Simran Khanuja, Sandipan Dandapat, Anirudh Srinivasan, Sunayana Sitaram, and Monojit Choudhury. Gluecos: An evaluation benchmark for code-switched NLP. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 3575–3585. Association for Computational Linguistics, 2020.

[3] Bidisha Samanta, Sharmila Reddy, Hussain Jagirdar, Niloy Ganguly, and Soumen Chakrabarti. A deep generative model for code switched text. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5175–5181. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[4] Ishan Tarunesh, Syamantak Kumar, and Preethi Jyothi. From machine translation to code-switching: Generating high-quality code-switched text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3154–3169, Online, August 2021. Association for Computational Linguistics.

[5] Jitao Xu and François Yvon. Can you traducir this? machine translation for code-switched input. *CoRR*, abs/2105.04846, 2021.

[6] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *ArXiv*, abs/1904.09675, 2020.