Design Document: Weakly-supervised person name transliteration from Latin script to other writing systems trained on Twitter data.

Greenland Yu
a1740184@student.adelaide.edu.au
The University of Adelaide
Australia

Lingqiao Liu
lingqiao.liu@adelaide.edu.au
The University of Adelaide
Australia

Matt Lowry
Fivecast
Kent Town, Australia
matt.lowry@fivecast.com

Jason Signolet
Fivecast
Kent Town, Australia
jason.signolet@fivecast.com

1 Problem Introduction

Transliteration in general is the process of converting a word from one writing system to another, this is done by swapping characters in one writing system to another in a predictable manner. The purpose of transliteration is to unambiguously represent the characters from one writing system to another. Standard transliteration systems for many script pairs exist, however, when individuals transliterate, results may differ from these official systems. Person name transliteration is the transliteration process applied to a person's name. Back-transliteration is the reverse of transliteration, trying to work out the original characters that has been transliterated into a different writing system. The process of back-transliteration poses more problems than normal transliteration as there is usually only a single correct back-transliteration of a word. Additionally, information from the source language is lost in the process of transliterating. This is because in the act of transliterating, the language you are transliterating to might not contain the means of expressing the sounds of the source language.

An example transliteration from Chinese characters (logogram writing system) to English (alphabetical writing system) for the name "余地" would be "yudi". The back transliteration of "yudi" to Chinese characters could have multiple different results such as "余底" or "芋地" or "芋底", this is due to the loss of tonal sounds in the initial transliteration.

To better understand transliteration, it is useful to understand phonemes and graphemes. Phonemes are the smallest unit of speech, while graphemes are the smallest unit of the writing system. For example the "a" in "ant" and the "a" in "baby" are pronounced differently in English. Even though "ant" and "baby" share the grapheme "a", the phoneme to pronounce the letter "a" is different. The standard systems that transliterate between writing systems rely heavily on graphemes and phonemes.

Due to the advent of social media, there is now an abundance of multinational social media users that are at times forced to use Latin characters to express their names; Twitter is a prime example. To create a Twitter account, user must enter a screen name that can be in any writing system under UTF-8. However, users are forced to input a username that has to be in Latin script, forcing the user to self-transliterate their screen name. We can take advantage of this by scraping Twitter user data to create a weakly labelled data set in which we can form a language model. Note that this data set is a representation of how people would informally transliterate their own name. So while we can describe this data set as weakly labelled (there are bound to be errors when self transliterating) we can use this data set to learn how informal transliterations are done.

2 Problem Motivation

2.1 Importance of Problem

With the continuation of globalisation, there is a great need for efficient and accurate machine translations so that people of different cultures can effectively communicate and express themselves to people of other different cultures. Person name transliteration is an important part of the wider machine translation field of study. Representation of a person's name in different languages is needed so that accurate and sensible machine translations between text documents can be achieved. An example of applied machine translation is Google Translate, a widely used service that applies neural machine translation techniques to translate between 109 different languages.

A concrete example of person name transliteration in use is the vetting procedure for foreign nationals. For anyone with access to sensitive data, that person must undergo security vetting to make sure their interests are aligned with the government agency. For foreign nationals their English name is usually a transliteration of their name in their country of origin. To perform accurate security vetting using these names, a back transliteration of their English name

must be performed to obtain the person name in the writing system of their country of origin.

2.2 Novelty

This project is an extension of a previous paper Mubarak et al. [10] that presents Arabic Twitter data as a novel data set to build a language model for transliteration between Arabic and English. This project will extend beyond Arabic Twitter data and aim to capture multi-lingual Twitter data to build language models for each language. Additionally, this project aims to evaluate the effectiveness of these language models built using Twitter data. The advantage of using Twitter data to train language models is that this data set represents transliterations done in an informal style and would more accurately capture how the average non-english speaking netizen would transliterate their own name. Additionally by conducting this project, we can evaluate Mubarak et al. [10] on just how useful of a data source Twitter really is.

3 Related Work

In this section we are going to survey work done in the field of machine translation and in particular person name transliteration. The main paper of focus is Mubarak et al. in which this project is based upon.

The basis of this project is Mubarak et al. [10] in which the author's present Twitter as a novel data source in which a parallel corpora can be obtained. The author's point out that in addition to having a sizeable amount of data, Twitter data is also easily obtainable. Additionally, in comparison to traditional person name parallel corpora, such as multilingual Wikipedia, this corpora reflects how real users would transliterate their own names to Latin script without a standard transliteration system. The authors demonstrate this by collecting 936,000 unique Arabic users from Twitter. A significant amount of effort in this study was the preprocessing of the Twitter data, this was to ensure that the dataset was valid and would be good candidates for statistical machine learning. Preprocessing of the data involved removal of titles from names and mapping of informal Arabic decoration characters to formal Arabic characters. Additionally, a similarity score was calculated between the Arabic and transliterated name. This similarity score was used to determine which dataset a name pair should go. The end result was three labelled datasets, one with name pairs of 100% similarity, one with 80% to 100% similarity and one with 70% to 80% similarity. The contents of these datasets were user name (Latin script), screen name (Arabic script) and geographical region. The reasoning for including geographical region was that name transliterations would vary region to region, including geographical region would mean that the researchers could infer a user's geographical region given their username. The datasets was evaluated by comparing against multilingual Wikipedia data, in which the combination of the Twitter dataset and Wikipedia dataset was used to build a character based translation model with Moses [6]. The model that used combined Twitter data showed a higher BLEU score [11] compared to the model built using solely Wikipedia data.

The second paper we are looking at is Prabhakar et al. [12] in which the researchers conducted a survey into machine transliteration and text retrieval methods. Their motivating factor in looking into machine transliterations is due to the advent of "Web 2.0" in which multi-lingual user generated data is increasing at a rapid rate, due to this there needs to be an effort to process this incoming data. The researchers come to the conclusion that "mining" approaches (where parallel corpora are mined from the internet and used to extract rules for transliteration) generally outperformed "generative approaches" (in which transliterations are governed by formal, bespoke rules for particular language pairs). The paper also discusses "fusion" approaches (a combination of the "generative" and "mining" approaches) to transliteration, in which researchers only show moderate success. The paper also explains transliteration basics in beginner friendly terminology.

Bilac et al. [3] explores the specific problem of back transliteration. They state the main challenges in back transliteration is that information is lost in the process of transliterating the person name. They survey previous work done under person name back transliterations and summarise approaches into two categories, "grapheme based" and "phoneme based". The researchers propose a new approach which is a combination of these two approaches. They achieve this by assigning weights to both models such that the weights maximise the accuracy when training the data. They evaluate their combined model in comparison to single model system and find that the combined approach outperforms the single model systems. During this project, when building the language model this approach can be considered.

An interesting problem to transliteration in general is the case when the origin of the name is neither from the source language or the target language. This particular case is addressed by Cohen et al. [4] in which they propose a method to transliterate names whose origin are not in either the source or target language. Their method relies on a monolingual resource list (such as a name list) and phonetic knowledge of the target language. The proposed method adds upon existing methods in the form of incorporating an origin specific transliteration table and origin specific n-grams model. These additions improved transliteration accuracy by 44% for names of Arabic origin from English to Hebrew. This specialised approach to transliteration could be incorporated into this project given there is enough time.

An approach to person name transliteration is through statistical translation models, Lee et al. [7] describe this process. Compared to previous models, statistical translation

models do not require a pronunciation dictionary or manually assigned phonetic similarity scores between source and target words. Instead a statistical translation model takes a noisy channel approach to word transliterations. The noisy channel approach utilises conditional probabilities and large amounts of parallel corpora data to generate a model. The authors describe this approach and show results on experimental data with 97.8% precision for person name transliterations from Chinese (logogram writing system) to English (alphabetical writing system). An implementation of a statistical machine translation engine is Moses [6], which was used in the Mubarak et al. [10] study.

Benties et al. [2] presents the TRANSLIT dataset, a large scale name transliteration resource. This dataset was formed by combining four different multilingual name corpora together which included JRC[5], Geonames[1], SubWikiLang[9] and En-Ar[13]. Additionally, Wiki-lang-all was compiled and added into this dataset by the researchers. Much of the work done during this study was the fusion of the different data sources, this involved removing duplicate names and processing all of the data to a standard format. This dataset would be interesting to compare to the dataset we plan to create in this project. Additionally, the background knowledge presented in this paper is a useful resource.

4 Project Goal and Challenges

4.1 Project Goal

The overarching goal of this project is to construct a transliteration pipeline that performs well over multiple languages and captures the unique subtleties that self-transliteration has. Figure 1 shows an outline of the training and evaluation pipeline. The goal for this pipeline would be to have well defined inputs and outputs for each component of the pipeline, this way changing the functionality of a component is straight forward. The input for the system would be raw Twitter data that has been extracted using a Twitter API key. The produced language model will attempt to capture how real users transliterate their own names into Latin script and present an alternative transliteration to the standard transliteration systems that already exist. This system should be easily adaptable, given Twitter data for another language, the system should output a language model for that language.

A large part of building this overarching system is having a robust and easily repeatable preprocessing system in place for new incoming Twitter data. The outcome of this would be a system that transforms Twitter data into parallel corpora for use in building a language model. The preprocessing step should include filtering out unnecessary data and cleansing user name and screen name pairs to those that are likely transliterations.

In the filtering stage, unnecessary data would involve getting rid of entries that are not tagged as the language we

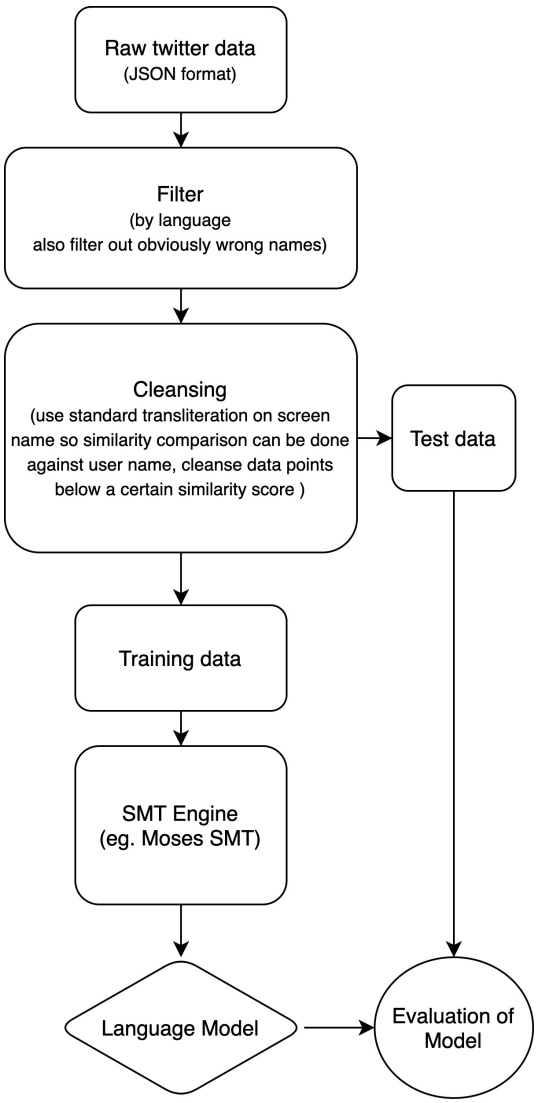

Figure 1. The training and evaluation pipeline

are interested in and also getting rid of entries that are obviously wrong such as names that consist only of emojis.

In the cleansing stage, to determine if a pair are likely transliterations, a transliteration of the screen name using a standard system can be performed and the Levenshtein edit distance [8] between that transliteration and user name calculated. Levenshtein edit distance is a way of measuring how similar two strings are based on the number of insertions, deletions and substitutions that have to be performed to transform one string to the other. If the Levenshtein edit distance is below a certain level for a name pair entry we can assume that the name pair are likely to be a transliteration and not useless data, we keep that name pair entry.

With our Twitter dataset the next step would be to build a language model using it. Most likely Moses [6] will be used as it is a proven open source statistical machine translation

toolkit. Other methods to build a language model will be explored while undertaking this project. Intrinsic evaluation of these models will need to be undertaken to determine how well it has learnt to transliterate.

Lastly, with the chosen language model, we can build the described pipeline. The technology stack and implementation of the pipeline is yet to be decided but python will most likely be used.

An extension to the overarching goal would be to have this system accept metadata found in the collected Tweets. Such metadata could include geographical region, Tweet creation date and whether the account is verified or not. Geographical region may effect name transliterations as regions will have differences in dialect even when speaking the same language. The Tweet creation date may have an effect on person name transliteration as more accurate transliterations would be from those with newer creation dates. Lastly, verified accounts should be viewed as a better source of person name transliterations compared to non-verified accounts and therefore should be weighted higher by the language model.

4.2 Challenges

The preprocessing stage of this project would constitute a significant amount of time. This is to ensure that the data set is valid and are good examples to train a language model on. An unavoidable challenge in preprocessing is a lack of language specific domain knowledge. For the many languages that Twitter users use, the variation between writing systems are vast.

Additionally, Twitter users are not required to enter their real full name for the screen name and user name fields when creating their account. These users will not be valid parts of the data set. Correctly filtering out these users in the captured Twitter data also poses a problem.

A solution to both of the presented challenges, ensuring that the data is valid without much language specific domain knowledge and filtering out useless data, is to use a standard transliteration system of the screen name and compare against the user name. If the Levenshtein edit distance [8] between the two is below a certain threshold we can confidently say that the user name is a transliteration of the screen name.

A potential challenge to this project would be that Twitter does not provide enough data, especially for languages that very few Twitter users use. A ramification of this would be that the language models built using this Twitter data would not have enough sufficient examples to properly learn the transliterations. Fusion of data from other resources may be needed to build these particular language models.

Another potential challenge would be if the language models created using the training data does not perform well on the test data. This may be due to a variety of reasons such as not having enough data to learn from and not stringent enough preprocessing of data. If this were to happen a review of the entire project process would need to occur, this way the root cause of the problem can be found.

5 Project Timeline

As of writing this Design Document, it is Week 4.

Week 4	Hand in Design Document
Week 5 and 6	Filter and cleanse Twitter data into valid person name transliteration pairs for multiple languages
Week 7 to 8	Investigate different methods of building language models. Test out transliteration accuracy of the data-set using these different language models.
Week 9	Start building person name transliteration pipeline
Week 10 and 11	Try and generalise over languages so the pipeline performs effectively over multiple languages
Week 12	Hand in Paper
Week 13	Hand in Video

References

[1] URL: http://geonames.org/.

[2] Fernando Benites et al. "TRANSLIT: A Large-scale Name Transliteration Resource". English. In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, May 2020, pp. 3265–3271. ISBN: 979-10-95546-34-4. URL: https://www.aclweb.org/anthology/2020.lrec-1.399.

[3] Slaven Bilac and H. Tanaka. "Improving Back-Transliteration by Combining Information Sources". In: *IJCNLP*. 2004.

[4] R. Cohen and Michael Elhadad. "Sideways Transliteration: How to Transliterate Multicultural Person Names?" In: *ArXiv* abs/1911.12022 (2019).

[5] Maud Ehrmann, Guillaume Jacquet, and Ralf Steinberger. "JRC-Names: Multilingual entity name variants and titles as Linked Data". In: *Semantic Web* 8 (Apr. 2016), pp. 1–13. DOI: 10.3233/SW-160228.

[6] Philipp Koehn et al. "Moses: Open Source Toolkit for Statistical Machine Translation." In: June 2007.

[7] C. Lee, J. Chang, and J. Jang. "Extraction of transliteration pairs from parallel corpora using a statistical transliteration model". In: *Inf. Sci.* 176 (2006), pp. 67–90.

[8] V. I. Levenshtein. "Binary Codes Capable of Correcting Deletions, Insertions and Reversals". In: *Soviet Physics Doklady* 10 (Feb. 1966), p. 707.

[9] Yuval Merhav and Stephen Ash. "Design Challenges in Named Entity Transliteration". In: *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 630–640. URL: https://www.aclweb.org/anthology/C18-1053.

[10] H. Mubarak and Ahmed Abdelali. "Arabic to English Person Name Transliteration using Twitter". In: *LREC*. 2016.

[11] Kishore Papineni et al. "Bleu: a Method for Automatic Evaluation of Machine Translation". In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, July 2002,

pp. 311–318. DOI: 10.3115/1073083.1073135. URL: https://www.aclweb.org/anthology/P02-1040.

[12] D. Prabhakar and Sukomal Pal. "Machine transliteration and transliterated text retrieval: a survey". In: *Sādhanā* 43 (2018), pp. 1–25.

[13] Mihaela Rosca and Thomas Breuel. *Sequence-to-sequence neural network models for transliteration*. 2016. arXiv: 1610.09565 [cs.CL].