Relationship extraction from Sanskrit Text

Sharath Prakash Kaushik - 7254517105 - spkaushi@usc.edu
Suhas Venkatesh Prasad - 9842999639 - suhasven@usc.edu
Sachin Subraya Pandit - 3141921859 - spandit@usc.edu
Meghana Purushothama Nayak - 2620870007 - mnayak@usc.edu

1)Introduction:

Relationship extraction involves the detection and classification of semantic relationship mentioned within a set of artifacts, typically from text. Extracting semantic relations between entities in text is a crucial step towards natural language understanding applications. Main entities in the text such as persons with specified relationships may only be mentioned implicitly and require efficient text processing techniques to be extracted.

Related work in this field includes [1], [2] and [3]. The paper of Efremova et.al, [1] presents a stochastic method for extraction of family relationships from Dutch notary acts using HMM and SVM. Santos et.al.[2] presents a rule based approach for extraction of relationships. Makhlouta et. al.[3] propose a method for extracting entities, events, and relations amongst them from Arabic text using a hierarchy of finite state machines driven by morphological features such as part of speech and gloss tags, and graph transformation algorithms. However, semantic relation extraction has not been extensively studied in Sanskrit, which motivated us to explore this domain.

In this project we extract family relationships from Sanskrit text by formulating the problem as a classification problem. We train and apply a Support Vector Machine to classify every name pair as a relation. We present a framework consisting of the following components: person name extraction, relationship descriptor identification, and pair-wise family relationship prediction.

We deal with typical challenges for real-world data collections such as the data quality problem, lack of training examples and imbalance in the dataset. The input to our method consists of Sanskrit text, and the output consists of pairs of person names with a predicted type of family relationship.

2) Materials - Corpus:

For the purpose of this project we use Ramayana as the corpus. The length of the text and repeating characters makes it a convincing choice for the corpus. We scrape [4] to obtain the raw corpus. There are six sections in the corpus and each section consists of nearly 80 chapters which corresponds to approximately 20000 shlokas(verses). The website from which we extract the

corpus has an English translation for the corresponding Sanskrit shlokas, this is helpful in analyzing the corpus and annotating the data.

3) Procedure:

The relationship extraction problem is formulated as a classification problem, where each class corresponds to a relation like father, mother, sister etcetera. The relationship extraction is done through a *SVM classifier*. We create data points, which contain features necessary for the classifier, in the below format:

Format: [<NAME1,NAME2>,<Relationship>,<Words in shloka with name pair>]

Example:[<कैकेयी, भरत>, <पुत्र>, <युष्मद् नी, आगतो, वीरो, युधाजित्, मातुल, युष्मद् श्रु, दशरथ, एतद् भरत, कैकेयी, सुत>]

[<Kaikeyii, Bharata>, <putra>, <Yushmadhni,aagatho,veero,yudhatith,maathul,yushmadh,shri,dasharatha,ethad, Bharata,Kaikeyii,suta>]

3.1) Annotation:

Named pairs are identified and their relationships are inferred based on our understanding of the corpus(Assessment of the annotation is explained later). The relationships are stored in the format *<Name1, Name2, Relationship>*. Additionally, we maintain a list of synonyms for each name and relationship. The following number of unique name pairs and its corresponding relationships are identified.

Relationship Number	Relationship	Number of unique name pairs	Relationship Number	Relationship	Number of unique name pairs
1	पति:(Pathihi/Husband)	18	6	शिष्य(Shishya/Student)	1
2	पुत्र(Puthra/Son)	21	7	श्वश्रू(Shwashru/Mother-in-law)	1
3	भ्राता(Bhratha/Brother)	14	8	मातामह(Mathamaha/Maternal- Grandfather)	1
4	भगिनी(Bhagini/Sister)	8	9	मित्र(Mithra/Friend)	1
5	कर्मकरी(Karmakari/Maid)	4	10	दौहित्र(Dauhithra/Grandson)	1

3.2) Pre-processing:

The data points are extracted from the raw corpus using the following steps:

1. We segment the corpus based on clear delimiters such as '||', which indicate separation of shlokas in the text.
2. For each word in a segment, we extract its root form through the morphological analyzer [5]. Due to the limitations of the tool, some words may not be processed, in which case the original form of the word is retained.
3. For each processed shloka having at least one name pair, we do the following:
    a. Identify and extract all the name pairs.
    b. For each name pair, check if a relationship exists between them using the annotated data.
    c. If a relationship exists in the above step, we create a data point for further processing.

3.3) Classification approaches:

The data points created in the above step are analyzed to understand their distribution in the corpus. Following are a few observations made from the analysis:

1. There is a great level of imbalance in the distribution of data for each class in the corpus. For example, married couples tend to occur a lot in the initial sections of the corpus and other relationships, say brother, tend to occur in the later sections.
2. The size of data available for each class vary drastically. For example, Married class has around 150 entries, Brother has 75 entries but Friendship has 7 entries.
3. A lot of name pairs identified in the corpus did not have a corresponding relationship in the annotation mapping.

These observations are significant as they help us decide on specific approaches to adopt for classification.

3.3.1) Binary vs Multi-class:

There are two possible ways to approach this classification, either as a multi-class classification - Relation1 vs. Relation2 vs. Relation3 (Father vs. Mother vs. Sister) or as multiple binary class classification - Relation vs. Not Relation (Father vs. Not Father, Mother vs. Not Mother etcetera).

From observations 2 and 3, we infer that data could be sparsely distributed for one class when compared to the other classes. This could lead to a potential issue for the multiple binary class classification approach. If there is a large amount of training data for the Not Relation class when

compared to the Relation class, most of the test data will be classified as belonging to Not Relation class. Hence, we will implement the classification as a multi-class classification.

3.4) Data division approaches:

The observations presented above post a challenge on how we can utilize the data points created to stipulate the classification problem,which in turn is necessary to obtain meaningful results. To overcome these challenges we implement two different approaches to divide the data in such a way that there is enough data for a given class in both the training and test corpus.

3.4.1) Splitting at corpus level - Approach I:

For each section in the corpus, 75% of the data is used for training and the remaining 25% is used for testing. However, since the relationships are not evenly distributed as seen in observation 1, it is possible that we could encounter an unseen relationship in the test data, thereby leading to an improper classification. Also, there is a possibility that a name pair is seen both in the training data and the test data. This does not make sense as we will then try to predict a relationship that is already known.

For the corpus we have considered in the project, there are 6 sections with each made up of around 80 chapters. So, the training set consists of 60 chapters(80*0.75) and the test set consists of 20 chapters from each of the 6 sections.

3.4.2) Splitting at class level - Approach II:

In this approach we segment the data at class level ie..,the data points obtained after pre-processing is considered class wise and for each class, the training and test data are segregated in the ratio of 3:1 after sorting its name pairs based on their count. However, while splitting the data, care is taken to ensure that data points do not belong to both training and test data set. Hence, there is a possibility that the split ratio is not strictly maintained between training and test data points. For example, let the distribution of data points for a particular class be as shown below:

Relationship:भ्राता

Pair:लक्ष्मण|राम(Laskshmana|Raama) count:116, Pair:वालि|सुग्रीव(Vaali|Sugreeva) count:1, Pair:भरत|राघव(Bharata|Raaghava) count:9, Pair:राम|भरत(Raama|Bharata) count:35, Pair:लक्ष्मणो|भरत(Laskshmana|Bharata) count:2, Pair:रावण|शूर्पणखा(Raavana,Shoorpanaka) count:1, Pair:भरत|शत्रुघ्नो(Bharata|Shatrugna) count:2

Count for relationship:166

The split made will be:

Train:

[लक्ष्मण|राम],[राम|भरत] =>151 count

[Lakshmana|Raama],[Raama|Bharata] => 151 count

Test:

[वालि|सुग्रीव],[भरत|शत्रुघ्नो],[सुग्रीव|वालि],[लक्ष्मणो|भरत],[रावण|शूर्पणखा],[भरत|लक्ष्मणो] =>15 count

[Vaali|Sugreeva],[Bharata|Shatrugna],[Sugreeva|Vaali],[Lakshmana|Bharata],[Raavana|Shatrugna],[Bharata|Lakshmana] => 15 count

Here, the training data points count exceeds 3:1 ratio because, as mentioned same name pair is not split between training and test dataset.

Ideally, there would be enough data such that random sampling is highly likely to result in a split where all the relations are represented in both training and test corpora; in this case a random sampling is sufficient. However, as seen in observations 1 and 2, it is not possible to find a single split of data for all relations, thereby making the above approach necessary for practicality.

It is noteworthy that a certain percentage of the training data is set aside as validation data.

The last step of the relationship extraction process is learning the model and classifying candidate pairs into the relationship classes. We apply and evaluate the designed technique using the Support Vector Machine (SVM) from the scikit-learn python tool.

4) Evaluation:

4.1) Annotation evaluation:

Inter Annotator Agreement: The annotation was performed manually by two people in the group. This annotated data was verified against the dictionary of characters and their corresponding relationships present in the website - [6]. The two sets of annotated data were compared and the annotations matched in more than 90% of the cases.

4.2) Relationship extraction evaluation:

As there is no previous work of this kind for Sanskrit text, there is no established baseline to compare our work. Hence we use a simple model with grammar of form $<NAME_1>$ ... $<RELATIONSHIP>$ .... $<NAME_2>$ to categorise each $<NAME_1,NAME_2>$ into a particular relationship. All the words in a shloka containing a name pair is analyzed to see if any of them is

an annotated relationship. If so, the name pair is categorized as that relationship. This is done for all shlokas in the test data.

The above method is used as baseline to evaluate our method.

5) Results:

We used the validation to fine tune the various parameters of the classifier, one such parameter is TFIDF which scales down the impact of tokens that occur very frequently in a given corpus and that are hence less informative than features that occur in a small fraction of the training corpus. This value was empirically determined with various validation runs on the training data.

The other features of SVM included a penalty parameter of 10 and also the use of a linear kernel. The high cardinality of our data set made linear kernel a suitable option for the classifier.

The tables below describe the precision, recall, f1-scores and accuracy(no. of correct classifications / total test samples) for the two approaches and their respective baseline for every relationship.

Approach 1:

Classes	Precision	Recall	F1-Score	Support
कर्मकरी (Karmakari / Maid)	0.00	0.00	0.00	1
पतिः (Pathihi / Husband)	0.33	0.36	0.34	14
पुत्र (Puthra / Son)	0.44	0.33	0.38	33
भगिनी (Bhagini / Sister)	0.00	0.00	0.00	0
भ्राता (Bhratha / Brother)	0.73	0.75	0.74	81
मित्र (Mithra / Friend)	0.00	0.00	0.00	2
Avg/Total	**0.60**	**0.59**	**0.59**	**131**

Accuracy:0.587786259542

Technique:Baseline:

Classes	Precision	Recall	F1-Score	Support
कर्मकरी (Karmakari / Maid)	0.00	0.00	0.00	1
पतिः (Pathihi / Husband)	0.00	0.00	0.00	14
पुत्र (Puthra / Son)	1.00	0.15	0.26	33
भ्राता (Bhratha / Brother)	1.00	0.09	0.16	81
मित्र (Mithra / Friend)	0.00	0.00	0.00	2
Avg/Total	**0.87**	**0.09**	**0.16**	**131**

Baseline evaluation: 0.0916030534351

Technique: Approach2

Classes	Precision	Recall	F1-Score	Support
कर्मकरी (Karmakari / Maid)	0.25	1.00	0.40	1
पतिः (Pathihi / Husband)	0.36	0.44	0.40	9
पुत्र (Puthra / Son)	0.25	0.28	0.26	18
भ्राता (Bhratha / Brother)	0.78	0.79	0.78	39
मातामह(Maathamaha/Maternal- Grandfather)	0.00	0.00	0.00	2
मित्र(Mithra/Friend)	0.00	0.00	0.00	6
Avg/Total	**0.51**	**0.55**	**0.52**	**75**

Accuracy:0.546666666667

Technique:Baseline

Classes	Precision	Recall	F1-Score	Support
कर्मकरी(Karmakari/Maid)	0.00	0.00	0.00	1
पतिः(Pathihi/Husband)	0.00	0.00	0.00	9
पुत्र(Puthra/Son)	1.00	0.28	0.43	18
भ्राता(Bhratha/Brother)	1.00	0.18	0.30	39
मातामह(Maathamaha/Maternal- Grandfather)	0.00	0.00	0.00	2
मित्र(Mithra/Friend)	0.00	0.00	0.00	6
Avg/Total	**0.76**	**0.16**	**0.26**	**75**

Baseline evaluation: 0.16

From the tables, we see that the classifier performs well for the relationship भ्राता/Bhratha/ Brother. From the analysis of the data points, we know that the data points for class भ्राता is uniformly distributed across the corpus. Thus, we see good F1-scores for this class in both the approaches indicating the importance of having an even distribution of data.

For each of the relationships मातामह/Mathamaha/Maternal-Grandfather and मित्र/Mithra/Friend, there is only one name pair observed in the corpus. Thus, we do not have enough data to train on, leading to poor performance of the classifier.

In both the approaches the classifier clearly outperforms the baseline model we have adopted as seen in the tables above. This strongly indicates that there are features much more powerful than the relationship name, which can be leveraged for the relationship extraction problem.

6) Discussion:

From the results seen above we can confidently claim that approaching the relationship extraction problem using supervised learning is a feasible option. In this instance of the problem where a SVM classifier is used against a Sanskrit corpus, the biggest challenge faced was handling the unevenness of relationship spread across the corpus. Although we have presented two different ways to handle the problem (making training-test split at corpus and class level), the corpus level splitting of data has the flaw of seeing the same name-pairs in both training and test corpus. This makes us consider its result with a level of skepticism as certain proportions of correct classifications made in this approach would be due to this factor. The second approach

although involves certain level of data tinkering, the results can be much more confidently claimed as the name pairs trained and tested on are clearly distinct.

Even though the overall performance of the classifier is around 55%, it significantly outperforms the baseline. Considering the dependency of identifying name pairs and relationship on the annotated data, we are quite confident that much better results will be obtained with larger set of annotated data. Although we have based our choice of classifier as SVM from the work of Efremova et.al.[1], perhaps much better results could be obtained using a much stronger classifier like Random Forest which is much more suitable for multi class classifications**.** Another exciting approach would be to use an unsupervised technique for relationship extractor in which case the emphasis placed on annotation and data sparsity would be greatly reduced.

In general, extracting semantic relations between entities is an important problem in natural language processing. Semantic relation extraction has not been extensively studied in Sanskrit, and we believe we have mentioned an elegant way to extract suitable information. The mentioned technique will sure be an encouraging factor for people interested in the topic.

7) References:

1. Towards population reconstruction: extraction of family relationships from historical documents - Julia Efremova et al https://dmm.anu.edu.au/popinfo2015/papers/2-efremova2015popinfo.pdf
2. Extraction of Family Relations between Entities - Daniel Santos et al http://www.inesc-id.pt/pt/indicadores/Ficheiros/9070.pdf
3. Arabic Entity Graph Extraction Using Morphology, Finite State Machines, and Graph Transformations - Jad Makhlouta et al
4. Corpus - http://www.valmikiramayan.net/
5. Lexeme tool - http://sanskrit.uohyd.ac.in/scl/
6. Relationship between characters in Ramayana - http://hinduism.about.com/od/epics/ss/The-Ramayana-Character-Map.htm#step2

8) Division of labor:

- Pre Processing - Sharath
- SVM Implementation & Feature Selection - Suhas & Sachin
- Evaluation & Baseline Implementation - Meghana.