# GSOC – 2020 Proposal for "Translating the whole Ur III corpus "

## Organization Information :

**Organization Name:** Cuneiform Digital Library Initiative (CDLI)

**Project Title:** Translating the whole Ur III corpus

---

## Profile Information :

**Name:** Himanshu Choudhary
**Email:** himanshuchoudhary_bt2k16@dtu.ac.in
**Github:** himanshudce (Himanshu)
**LinkedIn**: https://www.linkedin.com/in/himanshudce
**Location**: New Delhi, India
**Typical Working Hours**: 12 P.M. - 9 P.M. IST (UTC+05:30)

---

## ABSTRACT :

This project aims to build a full translation pipeline with the integration of NER (Named Entity Recognition) and POS (Part of Speech) tagging of Sumerian language and Post-processing of the translated English text to produce best and accurate results. The model should be built either using rule based or Neural based approach. The final pipeline including the Machine Translation System (Different Project) should be able to tag and translate the whole 1.5M Sumerian text perfectly.

**Possible Mentors:** Niko Schenk, Ravneet Punia.

---

# TABLE OF CONTENT

# Implementation Overview :

Data Pre-Processing

Named Entity Recognition [MODEL-1]

Pos-Tagging [MODEL-2]

Machine Translation System Improvement (Sumerian-English)
[MODEL-3]

Post-Processing of Translated Text [MODEL-4]

# Implementation Plan :

## 1. Dataset & Preprocessing

Here we will start our work on the raw data as given on CDLI github (https://github.com/cdli-gh/data) and it will be preprocessed after discussing with the mentors so that the NER and POS tagging models can be applied on the sumerian text. We have a very limited labeled dataset for the supervised learning as given on (https://github.com/cdli-gh/Sumerian-NER/tree/master/Raw/CDIL_morph_raw). Here both POS tag and NER are combined in a single column which needs to be separated to train our model. I will apply a three step iterative process as described below so further preprocessing will be done after discussing with the mentor.

## 2. Named Entity Recognition (Sumerian Language) [TASK-1]

For Named Entity Recognition Task I will extend the Research conduct by Niek Veldhuis (https://github.com/niekveldhuis/UrIII-names) to capture all the Named entities as described by ORACC version. Basically there are 16 categories for the named Entities.

| | |
|---|---|
| DN | Divine Name |
| EN | Ethnos Name |
| GN | Geographical Name* |
| MN | Month Name |
| ON** | Object Name |
| PN | Personal Name |
| RN | Royal Name |
| SN | Settlement Name |
| TN | Temple Name |
| WN | Watercourse Name |
| AN | Agricultural (locus) Name |
| CN | Celestial Name |
| FN | Field Name |
| LN | Line Name (ancestral clan) |
| QN | Quarter Name (city area) |
| YN | Year Name |

Since We do not have a larger labeled dataset to train the neural network model I will work on rule based and machine learning approach to tag with the appropriate named entity. The whole text passes through the 3 iterations and will be tagged accordingly.

## A.] Search Based [ Iteration 1]

Some of the named entities are limited, so we can creat the list of these named entities along with their limited names in a list and while iterating over the text these can be searched accordingly.
As per my knowledge the following named entities are limited -

**Royal Names (symbol - RN)**
There are only 5 royal names: UrNammak, Šulgir, AmarSuen, ŠuSuen, and IbbiSuen.

**Month Names (symbol - MN)**
https://cdli.ucla.edu/tools/ur3months/month.html

**Year Names (symbol - YN)**
http://cdli.ox.ac.uk/wiki/doku.php?id=year_names

**Divine Name (symbol - DN)**
http://oracc.museum.upenn.edu/amgg/listofdeities/index.html
(Can also be determined by the symbol {d} in front, which i will discuss in Rule-Based-Approach)

**Object Name (symbol - ON)**
http://cdli.ox.ac.uk/wiki/doku.php?id=the_one_hundred_most_important_cuneiform_objects
These are the top 100 most common objects of sumerian time. Though it will not cover all the objects, if any word is among these objects it can be tagged with ON for sure. I think It is a good start for the first iteration.

**NOTE -** It may be possible that some other names are limited as well, I will create list of these names such as I have commited the list of months names on github (https://github.com/cdli-gh/Sumerian-NER/blob/master/Cleaned_Dataset/months_ner.csv) and the words which will match will be tagged with these NER during the first iteration. It may be possible that for some particular entity it will not cover all the names such as Divine names which can be tagged in further iterations, but if it does it will surely be a god (divine) name.

## B.] Rule Based [ Iteration 2]

Some Named Entities can be determined by analysing the morphological syntax and applying some set of rules. Rules can be created if we are sure for some particular Named entities such described in (https://github.com/niekveldhuis/UrIII-names/blob/master/UrIII_name-authority.ipynb) as -

**Personal Name (Symbol - PN)**
In BDTNS names start with a capital or with {d}.

**Settlement Name (symbol - SN)**
Settlement names are followed by the determinative {ki}. So we can write the code and if the word is followed by/in between {ki} it can be tagged as Settlement Name.

**Divine names (symbol - DN)**
Divine names are preceded by the determinative {d}.

Since I have a limited knowledge of Sumerian language I am not very well aware of other rules. I will create the rules after discussing with the mentors and other experts.

**NOTE -** If the word is already tagged in the first iteration. It will not be tagged in this iteration (Second Iteration) as the tagged named entity is sure and fixed in the first iteration.

## C.] Machine Learning/Deep Learning Classifier [Iteration 3]

For the remaining named entities which do not have any set of list or rules, I will train a classifier with the remaining entities as labels words(Embeddings if available) of the sumerian text as the training dataset which is available on github (https://github.com/cdli-gh/mtaac_gold_corpus/tree/workflow/morph/to_dict and https://github.com/niekveldhuis/Sumerian-network/tree/master/bdtns_raw_data ). If there is more annotated data we can use them as well.

**Classification Model -** In this dataset we have a column XPOSTAG where the word is tagged with NAMED ENTITY and the POS tag. There are around 2000 words in the dataset. Since both named entity and POS tag are combined in one column we need to seperate them in two columns one with NER and other with POS tag. The classifier will contain the classes with the remaining entities and one class with **NOT NAME.** In the dataset XPOSTAG column if the word is not tagged with noun(N) I will add Not Name there.

To train the classifier there are various Models such as -
i.] Deep Learning classification Model
ii.] Decision Tree
iii.] SVM
iv.] Logistic Regression
v.] XGboost and many others.

I will try all the models and will use which will perform best on our dataset.

So, in the final iteration (Iteration 3) I will use our trained model to predict the class of the word. If the word is tagged already or from the NOT NAME class we will move forward otherwise we will tag the word according to the predicted class name.

# 3. POS (Part of Speech) Tagging [TASK-2]

For the POS tag I will use the two step Iteration. The first one will be rule based and the other one is the same as above (Machine learning classifier). In case of POS tag there are many symbols but very less labeled data and information is available to create rules and to train the models. There are various tags as -

AJ adjective (including statives)
AV adverb

| | |
|---|---|
| NU | number |
| CNJ | conjunction |
| DET | determinative pronoun |
| J | interjection |
| N | noun (including statives) |
| PP | possessive pronoun |
| V | verb (including infinitives, marked with EPOS ‘N and gerundive GW) |
| IP | independent/anaphoric pronoun |
| DP | demonstrative pronoun |
| MOD | modal, negative, or conditional particle |
| PRP | preposition |
| QP | interrogative pronoun |
| RP | reflexive/reciprocal pronoun |
| REL | relative pronoun |
| SBJ | subjunction |
| XP | indefinite pronoun |

## A.] Rule based [ Iteration 1 ]

In the iteration one we can use the rules for some of the POS tags.

**Noun (N) -**

If the word is already tagged with any NER symbol. It can be easily tagged as NOUN(N).

**Number (NU) -**

If the word are followed by a number and are in the form such as -
2(u)
5(disz)
2(u)
NUMB
These can be tagged as NU

If there are other rules for some specific POS tags, these can be added in this part and can be applied to the iteration 1.

## B.] Machine Learning/Deep Learning Classifier [ Iteration 2]

To classify the other pos tag entities I I will train a classifier with the remaining entities with the same dataset as described above - (https://github.com/cdli-gh/mtaac_gold_corpus/tree/workflow/morph/to_dict and https://github.com/niekveldhuis/Sumerian-network/tree/master/bdtns_raw_data ). If there is more annotated data we can use them as well.

**Classification Model -** In this dataset we have a column XPOSTAG where the word is tagged with NAMED ENTITY and the POS tag. There are around 2000 words in the dataset. Since both named entity and POS tag are combined in one column we need to seperate them in two columns one with NER and other with POS tag. The classifier will contain the

classes with the remaining POS tag entities. Here we do not need extra NOT POSTAG class as the word must be something out of those pos tag entities.
To train the classifier there are various Models such as -

i.] Deep Learning classification Model
ii.] Decision Tree
iii.] SVM
iv.] Logistic Regression
v.] XGboost and many others.

I will try all the models and will use which will perform best on our dataset.
So, in this second Iteration I will use our trained model to predict the class of the word. If the word is tagged already we will move forward otherwise we will tag the word according to the predicted class name.

## 4. Neural Machine Translation [TASK-3]

### A.] Tunning Current Model architecture (GSOC-2019)

We already have a Neural Network-based Encode-Decoder architecture for English-Sumerian Machine Translation using 10k parallel corpuses. Various architectures are explored such as -
**a.]** Unidirectional LSTM encoder-decoder neural network with word embeddings and Bahdanau/Luong Attention.
**b.]** Bi-directional LSTM encoder and Unidirectional decoder architecture with word embeddings and Bahdanau/Luong Attention.
**c.]** Transformer based (self-attention, google MT architecture) Neural network architecture with parameters as described in OpenNMT-py (https://opennmt.net/OpenNMT-py/FAQ.html#how-do-i-use-the-transformer-model).

**TASK -** As the transformer model is very sensitive to hyperparameters so it becomes necessary to try various options in order to get the best BLEU score.

### B.] Machine translation using Byte-Pair-Encoding(BPE)

BPE is a data compression technique that replaces the most frequent pair of characters/signs in a sequence. BPE helps in the suffix, prefix separation, and compound splitting which in our case can be used for creating new and complex words of Sumerian and English language. BPE performs very well for low-resourced complex structured languages. I have previously demonstrated in my research paper how well Byte-pair-encoding along with Pre-trained BPE embeddings and Multihead self-attention (Transformer model) perform for morphologically rich Indian Languages (Himanshu Choudhary, Ref.LREC-2020 conference **(submission ID 149)** (Accepted Papers) ).
Implementation - https://github.com/himanshudce/MIDAS-NMT-English-Tamil

**Neural Machine Translation using Byte-Pair-Encoding (Task)-**

a.] Bi-directional LSTM encoder and Unidirectional decoder architecture with Byte-pair-Encoding (trained/pre trained embeddings) along with Bahdanau/Luong Attention.

b.] Transformer based (Multi-head self-attention) Neural network architecture along with Byte-pair-Encoding (trained/pre trained embeddings).

### C.] EXTRA - Copy Attention Mechanism with NMT

Copy Attention(Copy net) is a great way to train the model if certain segments in the input sequence are selectively replicated in the output sequence. For example in Sumerian-English translation if there are some words of sumerian language which are directly used in the english output we can use copy attention so that model can learn efficiently. I have observed from the datset that many sumerian names and and some other words are directly being used in the English output such as -

| Sumerian-Input | English-reference |
|---|---|
| **numb** icah | **numb** of lard |
| **lunincubur** idab | did **lunincubur** accept |
| mu **ludingirake** | since **ludingirake** |
| Etc. | |

So we can use copy attention to make our model more efficient. It can be used directly with any Neural Network Model. Even OpenNMT-py provides the direct option to use this.( https://opennmt.net/OpenNMT-py/options/train.html#Generator )

## 5. Post-Processing of Translated Text [TASK-4]

Since after using Byte-Pair-Encoding we do not get any unknown word as BPE can create any word from the characters. We can use Language modeling to make the translation more fluent which is used in Unsupervised Machine translation.

### Language Modeling

Language modeling is used to create more fluent sentences from the Noisy sentence. It is improved by making local edits using a language model that has been trained on lots of monolingual data to score sequences of words in such a way that fluent sentences score higher than ungrammatical or poorly constructed sentences. This technique is used in unsupervised neural machine translations.

The **BERT Language Model** is Implemented by Facebook and the full code is publicly available.

Source -
https://github.com/facebookresearch/XLM#i-monolingual-language-model-pretraining-bert

# Libraries and Framework:

- Stanford Core NLP, NLTK,Scikit-learn and Pandas will be used for all preprocessing and machine learning tasks.

- OpenNMT Toolkit ( https://github.com/OpenNMT/OpenNMT-py ) will be used for Machine Translation implementation (Model 3). The toolkit provides almost all the options to customize the training process. For BPE we can use my own github repo (https://github.com/himanshudce/MIDAS-NMT-English-Tamil)

- For BERT Language Model we need PyTorch (for implementation), and Fast-Text(for embeddings)

- All Supporting codes for other models are available on Github -
  1.] https://github.com/niekveldhuis/UrIII-names
  2.] https://github.com/cdli-gh/mtaac_gold_corpus/tree/workflow/morph/to_dict
  3.] https://github.com/wwunlp/sner
  4.]https://github.com/facebookresearch/XLM#i-monolingual-language-model-pretraining-bert

---

# Possible Challenges for Our UrIII Pipeline Task :

- We have a very small dataset to train the machine learning classification models in case of NER and POS tagging. Which can cause an overfitting problem so It is good to experiment with different architectures.
- Since I am not an Sumerian language expert so we may also need the guidance of an experienced person with this language who can help to make further NER rules and have an idea of the availability of the sumerian dataset such as list of god names(DN), Royal names(RN) etc.
- Sumerian is a pretty old language so longer sentences may face the problem of Out of Vocabulary (OOV), but Byte-pair-encoding can overcome these issues.

---

# Proposed Timeline :

- **31th March – 4th May (Application Review Period):** Getting familiarized with previous NMT code and Dataset, Brush up the NLP concepts, overview the OpenNMT Toolkit and Py-Torch

- **4th May - 1th June:** Communication with mentors and other CLDI participants of GSoC to know them and their projects, learn more about sumerian language and CLDI organization, setting up environments and Datasets for the coding phase.

- **Phase 1 (1st June – 29th June):**
  Coding Begins, implementation of MODEL 1, 2. Model 1 is regarding the Named Entity recognition of Sumerian language while MODEL 2 is for POS tagging.
  Sources - 1.] https://github.com/niekveldhuis/UrIII-names
  2.] https://github.com/cdli-gh/mtaac_gold_corpus/tree/workflow/morph/to_dict
  3.] https://scikit-learn.org/stable/
  4.] https://github.com/wwunlp/sner

- **Phase 1 Evaluation (29th June – 3rd July)**
  Submission of Results

- **Phase 2 (3rd July – 27th July):**
  Implementation of MODEL 3 and 4. MODEL 3 is almost same as the current model (Using OpenNMT-py) along with BPE and the parameter tuning-
  Sources -
  https://opennmt.net/OpenNMT-py/
  Github - himanshudce/MIDAS-NMT-English-Tamil

  Model 4, is the language modeling using BERT which is already available for english language.I will implement it for sumerian language using sumerian monolingual corpus.
  Sources-
  Github - 1.]
  https://github.com/facebookresearch/XLM#i-monolingual-language-model-pretraining-bert

- **Phase 2 Evaluation (27th July to 31th July)**
  Submission of Results

- **31st July – 24th August:** Comparing all Machine Translation architectures, discussion with mentors to deploy the final MT pipeline along with NER and POS tagging, translation of all 1.5M sentences using the final model.

- **Final Submissions (24th – 31th August):** complete the documentation, perform any other bug-fixes if needed, Active on Slack and ask for community feedback.

---

# DELIVERABLES:

## Prep Tasks:

1. Learn how to contribute at CDLI, understanding Sumerian language more closely
2. Exploring the dataset for better preprocessing, downloading and preprocessing the list of appropriately Named Entities (Months name, God names, year names, Royal names etc.)
3. Brushing up machine learning and NLP concepts(BPE, Fast-text, Py-Torch) and getting familiar with OpenNMT Toolkit

## Objectives:

1. Creating the best performing Machine Translation system architecture in the final pipeline to get the best possible result.

2. Using state of art techniques such as BPE and copy attention to improve the accuracy and for handling noisy and incomplete data.

3. Creating the list of all possible Named entities and Rules for the NER tasks.

4. Experimenting with different Machine Learning architectures to classify the Named Entities and POS tags with the limited dataset .

5. Calculating BLEU Score for every model with tuning variations for quantitative and qualitative analysis.

6. Adding a Command line interface and coordinating with the student developer of “Sumerian - English Machine Translation” and preparing full documentation for proper understanding to the best of our knowledge.

---

# About Me :

I am a final year B.Tech student, studying Mathematics and Computing Engineering at Delhi college of Engineering, New Delhi, India. Currently, I am working as a Student Mentor in Robotics and Machine Intelligence lab in my own college. Previously, I have worked as a Research Fellow in Industrial Technology Research Institute, Taiwan(https://www.itri.com/) and at IIIT Delhi(http://midas.iiitd.edu.in/). I have also published two research papers on Machine translation in two of the premier NLP conferences i.e. EMNLP (WMT-2018) and LREC-2020 respectively. I also got selected in the Erasmus Mundus Big Data management and Analytics program for my Masters (will study in ULB belgium, TU Berlin, Eindhoven University of Technology (TU/e)), sponsored by European Commision.

## Research Papers

I have Published two research papers in the conferences and one in the journal which is under review.
**1.]** Worked on a paper titled “Neural Machine Translation for English-Tamil” as a first author, my paper got published in World Machine Translation Workshop(**WMT-2018**) under **EMNLP conference**. (https://www.aclweb.org/anthology/W18-6459/)

**2.]** Recently one of my papers titled “**Neural Machine Translation for low resourced Indian languages**” also got accepted in **LREC-2020** main conference where I worked on BPE along with multihead self attention.(Submission ID 149)(Accepted Papers)

## Work Experience -

**1.] Industrial Technology Research Institute(ITRI),Taiwan** **May 2019-August 2019**

● Worked on an industrial collaborated project regarding Patent knowledge exploration and concept discovery

● Worked in order to reduce the manual efforts along with maintaining a comparable accuracy for analyzing and searching related patents

● Our work was based on Machine Learning and Natural Language Processing techniques to automatically detect inventions that are similar to the submitted application and mapping them to the Hierarchy

**2.] Indraprastha Institute of Information Technology(IIIT), Delhi** **May 2018-July 2018**

● Worked as a research Intern in IIIT Delhi, Developed speech recognition system for drones and built a Neural Machine Translation system which performed better than Google translator on Indian language pair (English-Tamil)

● Results produced in less training time with higher accuracy.

## Contribution in CDLI

I have started working on the Project and exploring the sumerian dataset. The BPE implementation of MT System is already on my own github which I will add soon on the CDLI github (https://github.com/cdli-gh/Machine-Translation). Other than that I have created Sumerian-NER repo on CDLI Github where I have already put the raw and final Dataset for the Named Entity recognition and POS tagging for sumerian language according to the ORACC version. Github - https://github.com/cdli-gh/Sumerian-NER

## Why am I the right person for this project ?

This project requires experience with Deep Learning and Natural Language Processing, specifically in Machine Translation. I have a very close experience related to these technologies and as a research student I am very much eager to contribute to this Organization. As I have mentioned earlier, I have published two research papers on Neural Machine Translation in two premier NLP conferences i.e. EMNLP(WMT-2018) and LREC-2020 and even currently I am working on an unsupervised MT for Indian Languages. Moreover, I gained similar working experience in IIIT Delhi and ITRI Taiwan. While working on these projects I got the experiences to work on Open Source projects, Git, Linux and other related tools and frameworks. Other than that I have also taken various popular MOOC courses as well -

● Machine Learning by Stanford University ( Andrew Ng )
● Neural Networks and Deep Learning by Stanford University ( Andrew Ng )
● Udemy - Data Science and Deep Learning in Python
● Machine Learning A-Z: Hand-On Python (Udemy)

● Data Science Natural Language Processing In Python
● Practical Deep Learning with Py-torch (Udemy)

## Achievements -

● Selected within top 22 students all over the world to receive Erasmus+ Scholarship for Master's in Big Data Management and Analytics program by the European Commision
● Selected within the top 26 research interns out of 3000 applicants all over the world in Industrial Technological Research Institute, Taiwan
● Full-Time Employment Offer from OYO (Software Engineer - 2020)
● Secured city rank 1 in National Science Olympiad
● Contributor in Precog Research group@IIIT Delhi and Member of MIDAS research Lab IIIT Delhi (http://midas.iiitd.edu.in/ )

## Skills -

● Deep Learning, Natural Language Processing, Data Analytics, Machine learning
● **Programming languages** – C++, C, Python, SQL
● **Frameworks** - Apache Spark, Keras, PyTorch, Tensorflow, Pandas, Numpy, Sklearn, NLTK
● **Tools** - Ipython/Jupyter Notebook, SQL, Ms-office, SPSS, OpenNMT-py and Seq2seq-TensorFlow (neural machine translation toolkits), Linux
● **CourseWork** - Data Structures and Algorithms, OOPs, DBMS, Operating Systems, Probability and Statistics, Stochastic Process, Linear Algebra, Financial Engineering, Graph Theory, Fuzzy Logic, Discrete Mathematics, Cryptography and Network Security

---

# Conclusion:

I have written the proposal to give an overview of how I will implement the various models along with the appropriate timelines. To the best of my knowledge I have explored all the possible NER and Machine translation systems and I believe these models have covered all the possible settings. All the models are implementable and all the deadlines are very practical, I can say so as I already have a very similar working experience in this field and I have explained every architecture along with their possible Sources/Codes.