****

ADA University

School of Information Technology and Engineering

Senior Design Project

**FINAL REPORT**

Project Title: ***“*Development of Part of Speech tagging System*”***

Authors:

1. [CE] Emil Jafarov

2. [CE] Sabir Zeynalli

3. [CE] Sayyad Akbarov

4. [CS] Shafahat Sardarov

Project Advisor: Dr Samir Rustamov

Baku, April 2020

Table of Contents

[I. INTRODUCTION 3](#_Toc38921243)

[Definition 3](#_Toc38921244)

[Purpose 3](#_Toc38921245)

[Project Objectives, Significance, Novelty 3](#_Toc38921246)

[Problem statement 4](#_Toc38921247)

[II. LITERATURE REVIEW 4](#_Toc38921248)

[III. DESIGN CONCEPT 5](#_Toc38921249)

[Alternative Solutions/ Approaches/ Technologies 5](#_Toc38921250)

[**Rule-based PoS Tagging** 5](#_Toc38921251)

[**Lexical-based PoS Tagging** 5](#_Toc38921252)

[**Transformation-based PoS Tagging** 5](#_Toc38921253)

[**Memory-based PoS Tagging** 5](#_Toc38921254)

[**Stochastic PoS Tagging** 5](#_Toc38921255)

[Detailed Description of Solutions/ Approaches/ Technologies of choices 6](#_Toc38921256)

[**Hidden Markov Model** 6](#_Toc38921257)

[Research Methodology and Techniques 7](#_Toc38921258)

[Architecture, Model, Diagram description 8](#_Toc38921259)

[Social and environmental impact 9](#_Toc38921260)

[IV. Implementation 9](#_Toc38921261)

[Hardware Design 9](#_Toc38921262)

[Software Design 9](#_Toc38921263)

[Essential Components of the Project 10](#_Toc38921264)

[Timeline 11](#_Toc38921265)

[Testing/Verification/Validation of results 11](#_Toc38921266)

[V. Conclusion 11](#_Toc38921267)

[Discussion of results 11](#_Toc38921268)

[Future work 11](#_Toc38921269)

[VI. REFERENCES 11](#_Toc38921270)

[Appendices 12](#_Toc38921271)

[Screen shots of the software interface 12](#_Toc38921272)

[List of Figures 13](#_Toc38921273)

[List of Tables 14](#_Toc38921274)

[List of Abbreviations 15](#_Toc38921275)

***Abstract -* The study aims to demonstrate analysis and implementation of machine learning algorithms in order to build part of speech tagger system for Azerbaijani language.Various methodologies for PoS tagging have been analyzed, and stemmer to simplify process of constructing labeled corpus in Azerbaijani and Hidden Markov Model algorithm have been chosen for tagger tool based on tagged corpus after discussion of several approaches. The paper indicates stemmer as a necessary step to create PoS tagger application for Azerbaijani language. After comprehensive testing process of labeling word groups with stemmer, it has been clarified that stemmer has successfully handled most of exceptions. Then HMM was applied for examination a middle-sized tagged corpus in Azerbaijani. The outputs of testing are expected to be adequate for Azerbaijani language.**

***Keywords*** – **part of speech tagging; natural language processing; hidden markov model; Azerbaijani part of speech tagger; Azerbaijani stemmer**

# INTRODUCTION

## Definition

Part of speech tagging is considered as one of the essential pre-processing steps in most of the Natural Language Processing systems, which labels every word in the given text with a particular part of speech. PoS tagging term has many well-known definitions in the computational linguistic literature. Martin and Jurafsky explains PoS tagging term in [1] as: "Part-of-speech tagging (or just tagging for short) is the process of assigning a part of speech or other syntactic class marker to each word in a corpus".

The entire process of PoS tagger includes a number of steps. In the first step of process, punctuation marks such as period, semicolons, commas etc. is detected and separated from words. Second step of the process is eliminating suffixes from words in order to get base form of words by using lemmatization and stemming. During the whole process this operation has been called stemming, as Azerbaijani is agglutinative language in which root of word does not change depending on their morphemes and therefore meanings of lemmatization and stemming are so similar in this kind of language. The goal of final step is labeling words with appropriate part of speech. In this step, Stemmer changes the letters of root of each word to uppercase and compares them with listed tagged words in the dictionary and tags words with proper part of speech.

There are eleven parts of speech in Azerbaijani language which divided into two categories: main and auxiliary. The main category has six parts of speech: noun, adjective, verb, adverb, numeral and pronoun and five auxiliary parts of speech are consisting of particle, modal adjunct, postposition, conjunctive and interjection. Furthermore, verbs can behave as several parts of speech in the sentence such as noun, adjective, and adverb by taking features of them but as a part of speech it should be called verb. In the Table 1., examples of parts of speech in Azerbaijani language and their translation in English are provided.

|  |  |  |
| --- | --- | --- |
| Parts of speech | Azerbaijani | Translation |
| verb | danışmaq | to speak |
| adjective | gözəl | beautiful |
| noun | maşın | car |
| adverb | dünən | yesterday |
| numeral | beş | five |
| pronoun | biz | we |
| particle | yalnız | just |
| interjection | bəh! | well! |
| postposition | əvvəl | before |
| conjunction | ancaq | but |
| modal | bəlkə | maybe |

Table1. Parts of speech in Azerbaijani language and translations

## Purpose

Most of Natural Language Processing applications use part of speech tagging system as a base stage. The main purpose of our study is creating a large corpus and developing tagging system for the Azerbaijan language. From our perspective, this corpus will be effective method to pave the way for NLP software tools such as translators, speech synthesis systems, parsing systems and etc. in the Azerbaijan. Furthermore, this research can draw attention of famous companies, investors which work on NLP applications, to the Azerbaijani language. It will make easier their job because our language has quite different grammatic rules compared with other languages. We as a team want to create comfortable ambience for the people who interested in Azerbaijani language.

## Project Objectives, Significance, Novelty

The objective of our project is implementing PoS tagger which assign grammatical tag to sub-sentential tokens. Such units are consist of words and symbols. We want to implement PoS tagger for Azerbaijan language with high accuracy and a reliable large-sized tagged corpus.

This system will play significant role in improving NLP field in Azerbaijan because it can serve a basis for further projects related to NLP pipeline. People can have access our system and corpus freely via internet use it for their further innovations. Also, our project will have contribution to the spreading of Azerbaijani language-based tools in global technology world because it will facilitate to learn and understand our language for the foreign NLP specialists.

The number of solutions to part of speech tagging has been revealed during many years, but none of them is related to Azerbaijani language. First and oldest method of tagging is Rule-based approach. In this approach, we tokenize words by using dictionary. If words have more than one part of speech, we use hand writing rules. Second most popular approach is Stochastic tagging. It is about finding probability of possible tags and it always need train corpus. In 1960s, Brown University developed corpus by using this technique and it was the first main corpus for the English. It had 70% accuracy. In 1980s, Hidden Markov Model was used Lancaster-Oslo-Bergen Corpus. This model could disambiguate words that used as noun and verb. Steven DeRose and Ken Church coded new algorithm in the 1987 which had accuracy over the 95% and called dynamic programing approach. For better estimation a table of triples were used by them.

To sum up, there are many approaches for part of speech tagging and tag set has an important role in almost all of them. We believe in that Hidden Markov Model is best fitting algorithm for our language because Azerbaijani is very similar to Turkish and Turkish tagging system uses HMM.

## Problem statement

Problem is very clear. Tagger algorithm takes text as an input and as an output gives text with each word assigned particular tags or tags. The most complex part of tagging system is distinguishing words that have a different part of speech and choosing correct form. For instance, there are homonyms that can have various meanings based on context like “don” which can be noun and verb. In addition to this, several words can change their meanings depending on stressed syllable such as “açıqlama”, “vurma” and etc. Throughout the years, several techniques are improved by computer scientist in order to come with a solution to these problems in many languages. Unfortunately, we have a few researches related to part of speech tagging in the Azerbaijan language. For English, Russian and most of widespread languages have been developed such systems however in Azerbaijan, we don’t have any system, tools, even corpus for part of speech tagging. Also, it has great impact on improvement of other NLP technologies. From our point of view, this project will benefit not only ADA community but also all Azerbaijani speaking individuals.

# LITERATURE REVIEW

Around the world the number of PoS tagging systems are developed by different countries. Unfortunately, there are a few attempts to have this kind of NLP technologies such as Dilmanc, a project lead by Ministry of Communication and High Technologies and a Hmm/Viterbi based PoS tagger project with small size dataset from the previous work from the students of ADA University in the Azerbaijan. We have researched and analyzed several local and global literatures related to stemmer and PoS tagger which motivates us to do this project.

Merialdo [2] who is the creator of one of the oldest PoS taggers, compared two taggers with part of speech tagged datasets and a corpus which is not tagged. In his study, he conducted two analyses by using triclass Markov model. One of them was Forward-Backward algorithm and another was Relative Frequency. The formula is consisting of two type of calculation: Viterbi and Maximum Likelihood. With help of Viterbi algorithm, we can evaluate sentences and for evaluation of words ML algorithm is used. After comparing results, author claims that training model with RF algorithm large tagged corpus is the best option to get high accuracy.

Brants [3] introduces a new tagging approach which is designated based on second order of Markov Model and called Trigram Tagger (TnT). This method taking into consideration the probabilities of trigram, unigram and bigrams from labeled dataset in order to predict potential word. After calculating of probabilities, it applies smoothing technique because preventing any reducing in the effectiveness of the model related to data sparsity. Next step is handling words which is not identified in the dictionary with probabilistic model. Also, author states in [3] capitalization is effective for HMM as: “Additional information that turned out to be useful for the disambiguation process for several corpora and tagsets is capitalization information. Tags are usually not informative about capitalization, but probability distributions of tags around capitalized words are different from those not capitalized.”. Outputs of the mentioned method shows that average accuracy is 96% and 97% and can change depends on size of corpus and language. Brants claims that by using from capitalization, smoothing, and detecting undetermined words it is possible to get more precise values.

Lovin [4] claims that using iteration and longest-match algoritms together in order to create stemmer will produces better outcomes than having only one. In this method, longest-match algorithm tries to detect longest suffix. After finding longest word ending it will remove it from words and it will reduce steps of stemming. This method needs to have additional memory to store file which filled with suffix combinations. Also, he offers to use iteration algorithm to get the roots of words by removing suffixes with recursion function till derivational suffix that forms stem. Lovin states that correction technique is an effective way to increase accuracy. In case of spelling exceptions occurs when root changing suffixes add to word’s root, this correction algorithm replaces consonants based on provided if statements. These rules that applied to tagging process can be improved by time in order to get better accuracy score.

As already known, Azerbaijani language belongs to Turkish language family, we reviewed various Turkish researches. Bolucu and Jan [5] are one of the Turkish researchers who studied unsupervised Bayesian method with HMM to decrease sparsity. There are several failures happens because of morphological structure of words. Derivational and inflectional suffixes can cause failures to tokenize words in corpus. They applied the model tagging and stemming jointly and noticed that accuracy is getting up. It shows better results whenever it applied together.

Dincher, Karaoglan and Kishla [6] worked on PoS tagger system for Turkish language in order to build informational retrieval system. They considered seven different suffixes to determine part of speech in the tagged corpus. In their work, HMM was used with closed lexicon. According to their testing results average accuracy is 90.2% for the. In their studies, it was proposed as:” Our results show that a PoS tagging task with lexicon of closed vocabulary is possible for an agglutinative language as in Turkish by the use of simply suffixes.”

Valizada [7] did one of the valuable researches about principle of part of speech tagger and stemmer in Azerbaijani. In his study, he prefers Rule-based approach because in his opinion, Rule-based method uses time and resources efficiently. Also, he thinks that for small corpus we can get better results by applying few rules for stemming and tagging. His claim is that accuracy with Rule-based approach can be approximately 95%. He used Viterbi and HMM algorithms for tagging together and got 80% accuracy score by using database with 50000 words in Azerbaijani language.

Mustafali Ali, Sadigov Ziyaddin, Mollayev Rasim and Mammadov Samir [8] SITE 2018 students of ADA University under the supervision of Dr. Rustamov created PoS tagger for Azerbaijani language and published paper about it in the 8th International Conference on Intelligent Text Processing and Computational Linguistics. According to their record they worked on 2 main components one of them is stemmer another HMM and Viterbi algorithm. In the first step, stemmer stems word and tokenizes word with certain tag taken from dictionary. Secondly, program calculates transition matrix by using bigram HMM based on tagged corpus and Viterbi algorithm tags words sequentially. Accuracy of their program is 90% with corpus which is consist of approximately 3000 sentences and uncleaned dictionary.

# DESIGN CONCEPT

## Alternative Solutions/ Approaches/ Technologies

Part-of-speech tagging is the process of tagging a word in a corpus (text) with corresponding part of the speech. Part-of-speech tagging sometimes called. The tagging process of the word must be matching with its particular part of speech based on both its definition and its place in the sentence. Briefly, PoS tagging is the identification of words as nouns, verbs, adjectives, adverbs, etc.

Ambiguity is most common problem with PoS tagging as the word can have a few meanings based on its context and definition, therefore, a great number of different approaches of part of speech tagging have been created and used in different occasions. Some languages are more suitable for certain solution and others do not support the same technique. There are parts of speech tagging methods such as manual rule-based tagging and more advanced techniques called PoS taggers based on transformation and probability.

## **Rule-based PoS Tagging**

Rule-based PoS tagging is one of the oldest techniques of tagging. These taggers mostly use dictionaries for possible tags while tagging each word. If the word has more than one potential tag, at that point rule-based taggers use manually written standards to recognize the right tag. Disambiguation can likewise be acted in rule-based labeling by dissecting the phonetic highlights of a word alongside its former just as following words. For instance, assume in the event that the former expression of a word is article, at that point word must be a noun. In Rule-based PoS Tagging the information is always coded in the form of properties and rules. Moreover, these rules are limited by 1000.

## **Lexical-based PoS Tagging**

This method is about assigning the word with the tag which appears the most frequently with the word in the text corpus. The above-mentioned approach can be quite effective, depending on the text corpus and the language itself. The text corpus for this method requires to be much bigger. When combining these two methods, one can increase the accuracy significantly.

## **Transformation-based PoS Tagging**

Transformation-based Tagging is additionally called Brill labeling. It is an example of the Transformation-based learning (TBL), which is a rule-based calculation for programmed labeling of PoS to the given content. TBL, permits us to have semantic information in a discernible structure, changes one state to another state by utilizing transformation rules. It draws the motivation from both the past clarified taggers − rule-based and stochastic. If we see likeness between rule-based and transformation tagger, at that point like rule-based, it is also based on the standards that determine what tags should be appointed to what words. Then again, if we see comparability among stochastic and transformation tagger, at that point like stochastic, it is AI system in which rules are consequently actuated from information

For better comprehension of TBL, we can make a relationship to sculpting. Think about a stone worker chipping away at a human bust. At first, he has a cuboid bit of alabaster. At that point he, scratching off liberal bits of alabaster, attempts to cause the figure to look like a human head, without indicating face highlights, hair, and so on. Next, he goes further in details – he scratches a lot of littler pieces from the model to get a worthy likeness to a human (for the present, it, in any event, needs to remind a human head). Over the long haul on, the artist utilizes a littler blade, and deals with littler regions, for example, nose, eyes, hair, ears, and so forth. After difficult work, the worn-out stone worker at last gets an exact figure of a human with wanted highlights, expounded to the littlest detail. TBL works likewise to that of the artist. Initially, TBL takes the broadest conceivable rule from accessible rundown of tagging rules and tags the content. In this way, it searches for an increasingly explicit standard that will marginally change the tags made by the past, broadest principle. The procedure goes, until the content is tagged enough precisely.

## **Memory-based PoS Tagging**

Memory based PoS tagging is a supervised tagging which learns inductively from the samples. Examples are represented as a vector of feature values with an associated category label. Usually, in this type of PoS tagging the training set is presented to the classifier and then added to the memory. The approach is based on assumptions on direct reuse of stored trainings rather than on application knowledge which is abstracted from experience.

## **Stochastic PoS Tagging**

The simplest stochastic taggers disambiguate words based solely on the probability that a word occurs with a particular tag. In other words, the tag encountered most frequently in the training set with the word is the one assigned to an ambiguous instance of that word. The problem with this approach is that while it may yield a valid tag for a given word, it can also yield inadmissible sequences of tags. Stochastic taggers use both contextual and morphological information, and the model parameters are usually defined or updated automatically from tagged texts. These taggers are preferred when tagged texts are available for training, and large tag sets and multilingual applications are involved. In the case where additionally raw untagged text is available, the Maximum Likelihood training can be used to estimate the parameters of HMM taggers.

## Detailed Description of Solutions/ Approaches/ Technologies of choices

When deciding which technique to use, we came to a decision to combine some of the above-mentioned methods to make the program much more efficient and accurate. As it was said before, it is obvious that a gigantic dataset will be required. To ease the way of getting the tagged text corpus, we decided to use rule-based and lexical-based methods to tag the text instead of us, at least for the sake of some percentage of accuracy. After getting some percentage tagged with the help of these methods, the remainder would be tagged manually. There are a great number of good working HMM in the open source. Taking one as the sample, slightly changing the functionality and using it in the future was our decision, because focusing on proper dataset is priority.

With our project, we believe that the work we do now will be very useful and helpful in future for all-natural language processing tools and programs. The part of speech tagging master plays key role in language processing in almost all widespread languages worldwide. It can also be helpful in boosting the popularization of learning and using of Azerbaijani language, not only in Azerbaijan but in other countries as well. A great number of digital language learning facilities and opportunities can be developed on the basis of the accurately working part of speech tagger.

## **Hidden Markov Model**

Using probability in tagging errands is viable as it quantifies how likely it is for a word to be a sure part of speech dependent on insights got from a huge corpus. This technique creates very great outcomes on the grounds that syntactic attributes of a language is thought about when performing task, for instance, in specific dialects, verb is gone before noun while in others, it is pronoun rather than noun. The model that depends on probability of states which on account of tagging are tags and probability of progressing between these states is called Hidden Markov Model. The working standards and formalization of HMM in PoS labeling with models for English language is depicted in detail and plainly in figure 3. During the project, this depiction is acknowledged as a rule to comprehend and actualize HMM for PoS tagging effectively.

HMM for PoS tagging depends on the Bayesian inference which thus is a technique in measurable deduction and utilizations Bayes hypothesis to ascertain the probability of an occasion given another occasion. PoS tagging can be drawn closer as a classification issue, as the words in a given book are ordered into various classifications of tags. It is a sequence classification problem in because of the fact that the arrangement of tags is considered about as well while tagging.

To know the rules based on which HMM works, let us take an example and proceed. For instance, if we have a sentence “Karantin rejimi mayın dördünə qədər davam edəcək.”, which translates to “Lockdown will continue till fourth of the May.” in English, how does HMM calculate the most likely sequence of tags? According to Bayesian inference, all possible sequence of tags needs to be considered and out of these sequences, the one with the highest probability is chosen as the best sequence. The tag sequence of which the probability is the highest is chosen as:

This function chooses the sequence of tags that has maximum probability given sequence of words. We use Bayes rule for calculating this probability. The idea behind Bayes theorem is to change a probability which needs other obscure probabilities to be known into a connection of different probabilities which are known. The equation of Bayes theorem is formulized as:

which becomes:

When applied to the current problem which we have. The denominator of the formula shows the probability of across with a chosen word in a corpus. The probability does not change neither the number of specific word nor the corpus size, therefore, the denominator can be skipped, and we get the formula:

After simplifying calculations, HMM tagger considers the probability of a given word in a corpus doesn’t depend on other words but depends on only itself.

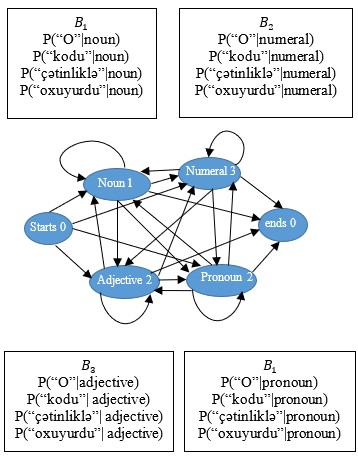
Taggers can be categorized by their function of assigning the word by their positions. Unigram tagger doesn’t consider the probability, therefore, there is no sequence in this case. In this project the bigram and trigram tagging methods are used which consider the previous tag also while tagging the next word. In bigram tagger, we calculate probability by multiplication of probabilities of tags given the previous tags to them.

This bigram tagging makes the work more accurate as Azerbaijani language is very rich with homonym words thus the tagger can decide the tag of the word by looking tag of the previous word. By this fact we obtain that, the tagger can choose whether the word “tut” is verb or noun by checking whether the word before the “tut” is adjective or noun. counts how frequently becomes visible in the text corpus and out of these appearances, how many are 𝑡𝑖. By the results, we calculate the probability.

There is also another method we used which called trigram tagging. In trigram tagger we calculate probabilities based on second order Markov models which considers triple of consecutive words.

HMM has two type of probabilities:First one is transition probabilities and observation likelihoods and the second one is word likelihoods. These probabilities are compacted into two matrices(A and B). In Figure 1 you can see the graphical representation of the transitions and states and in Figure 2 you can see hidden tags of the Hidden Markov Model and also transition probabilities between these tags which are denoting probabilities. Markov chain shows hidden states and observation of probabilities. Each word is tagged with a tag that is connected with a list of probabilities.

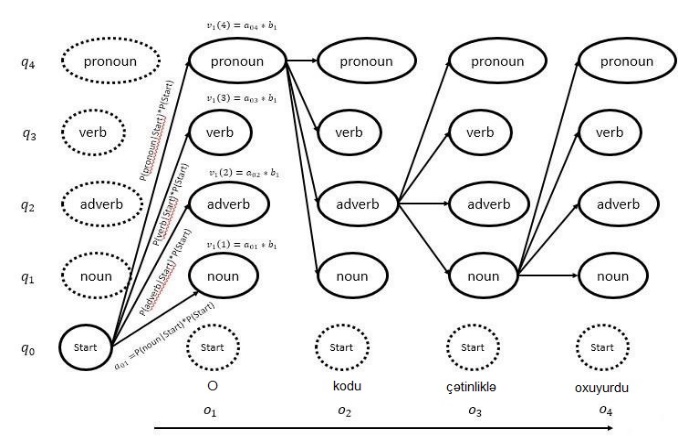
**Figure 1.** The Markov chain depicting hidden states (tags) of the HMM as well as transition probabilities between them denoting prior probabilities



**Figure 2.** The Markov chain depicting hidden states (tags) of the HMM as well as B observation likelihoods of words. Each tag is associated with a list of probabilities, one for each word.

Decoding is also one the most important parts of HMM model. Decoding means finding the hidden variable sequence, and while tagging, tag sequence. Viterbi algorithm is most useful decoding algorithm while using HMM.

The Viterbi algorithm is a dynamic programming algorithm that is used to find approximate sequence of the hidden states which called Viterbi path. The Viterbi algorithm is used not only in PoS tagging, but also in speech recognition systems. In Figure 3. The working process of the Viterbi algorithm is shown.



**Figure 3**. Working process of Viterbi algorithm.

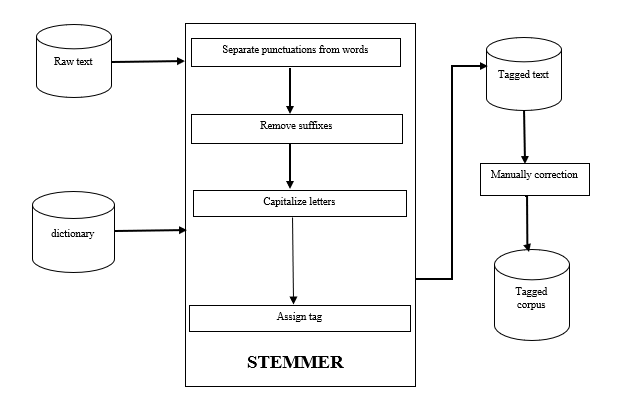
## Research Methodology and Techniques

We have used a lot of techniques for starting to our researches. First of all, we interviewed a lot of students and professors which are very close or directly connected with linguistics. This part was very crucial because Azerbaijani language is more difficult and trickier rather than English. For this issue our first goal was to understand our language in a way that would help us while making the technical decisions in future. The knowledge of linguists let us learn about difficulties and tricky moments of Azerbaijani language. While choosing the people for our interviews we also concentrate on timing as the timing part was also very important value because we were on the first steps of our project. We spent approximately two weeks with interviewing linguists. After learning all potential future problems and difficulties of our language we took our second step and start our qualitative research methods about technologies and approaches for our project. To find out the most suitable way or approach to our problem we asked graduated students of SITE about their experiences and difficulties while working with Natural Language Processing. As they had a big experience with NLP their answers were very beneficial for us while finding the best way to our solution. After finishing our interview part, we started to make a research on internet as we had answers both for linguistics and technical part, however, our aim was to find the way that would help us both with linguistic and technical part. After ten days of research we as a team made a meeting with our mentor and with unity decided to choose the method which all of us confirmed.

**Figure 4**. The research diagram

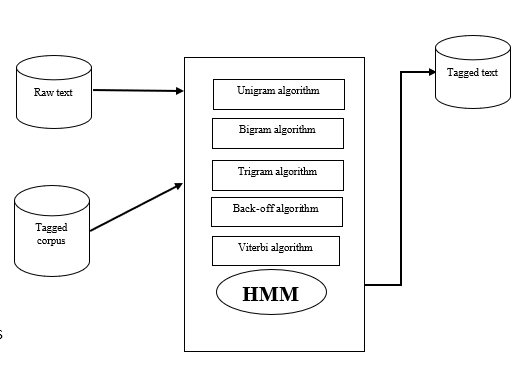
## Architecture, Model, Diagram description

After our qualitative research we gathered a lot of information for solving the problem with successful and suitable approaches, however, for implementing all of these data we needed to write and correct algorithms which we had beforehand. For this issue, we decided to choose the Python programming language which is very useful and user-friendly programming language. We had a few reasons for coming to this decision. First of them is Phyton’s structure which is very easy and for leaning and using and well-integrated system which can work with existing and independent IT infrastructures. Subsequent reason for choosing this programming language is usability of a data professional adept at handling any data-related query. Also, Phyton can support extensive support libraries and can easily done the all the commands which are related with Machine Learning an Artificial Intelligence. Final and the most important reason of choosing Phyton is its feature that can provide enhanced process control capabilities, object-oriented design as well as possesses strong integration along with text processing capabilities and Phyton’s unit testing framework which belongs to language itself. Despite all of these features the speed of the language is similar with its productivity. Because of all these skills and accessible platforms, we chose Python as our programming language.



**Figure 5**. The general architecture of the Stemmer.

Stemmer is the part of our algorithm which reduces words to their stem part by removing suffixes from the word. The data type which is given to the Stemmer is not needed to be filtered, the data can be the story, news or any scientific journal and no matter of the data length Stemmer begins to its work. First of all, Stemmer separates punctuations from the word and if needed shows them to us. Then it removes suffixes from each word of the given data individually. While removing suffixes it our Stemmer does it with certain rules. As in Azerbaijani language there are two types of suffixes inflectional and derivational, our algorithm deletes only inflectional ones because derivational suffixes transforms words from one to another part of speech. After this processes our Stemmer capitalizes given data in order to easily find the word from dictionary later. As soon as Stemmer finishes these actions, it directly goes to the dictionary and finds the stem part of each word in the data and tags them with suitable part of speech tags. The work of the Stemmer finishes in this point, and it gives the full tagged data with the normal status by adding deleted suffixes to the given words and notes “Undefined” the words which is not found in the dictionary.



**Figure 6**. The General Architecture of the HMM.

Before starting the HMM part, the data which is given from the Stemmer is checking manually by team member and after all errors and mistakes are fixed up, the tagged data becomes useful for giving to the HMM as input. HMM uses Bigram, Trigram, Back-off and Viterbi algorithms in for training and learning from the given tagged data. Bigram and Trigram algorithms are firstly used for learning the words. N-grams are tools for cutting out of sentence to consecutive words. Unigram, which is not used in our algorithm, takes each of word lonely, however bigram and trigram creates sets of consecutive words. Let’s take a sentence for example:

“Mən səyahət etməyi çox sevirəm.”

* Unigram – [Mən] [səyahət] [etməyi] [çox] [sevirəm.]
* Bigram – [Mən səyahət] [səyahət etməyi] [etməyi çox] [çox sevirəm.]
* Trigram ­– [Mən səyahət etməyi] [səyahət etməyi çox] [etməyi çox sevirəm.]

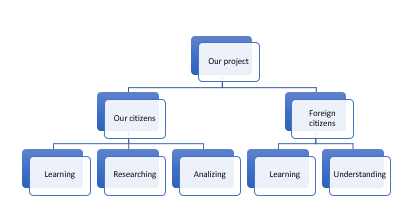
After these steps Back-off algorithm starts to its work. Firstly we use the output of trigram, if the result is not sufficent algorithm uses bigram for Backoff. Briefly, if trigram has zero counts, we make approximation with (N-1)-gram which is Bigram in this case and continue backing of till we have a data with counts of history.

Finally, our Viterbi algorithm calculates the most probable path which goes through the state transitions. It makes an outcome with the help of given trained corpus and some other data which HMM gains in previous steps. Finally Viterbi makes an array with collumns corrsponding to inputs and rows with possible states and fills these inputs based on transition.After these steps our algorithm learns from the given corpus and does automatically tagging according to its learning while giving a raw text to HMM.

## Social and environmental impact

It’s impossible to imagine the modern word without technology. The technology means a lot for our everyday life in terms of information, study, work, humor and so on. The impact of technology for social live is in a high level. İn our opinion, our project which is about not only technology but also language will have a great social impact on society.

There is undeniable bond between languages and individuals, moreover, NLP language is a proxy for human behaviour and human charachteristics. Because of all facts mentioned above, we think that our project’s effect will be very beneficial as it can help to school children, students, researchers and even for programmers who will start their researchs and projects about combination of language and technology. School children will efffectively use our application in order to check the correctness of their writings and researchers will easiliy get information about usage of language by analizing our work. Futhermore, our project will create the revival about digitalizing of our mother tongue. Hopefully, after a few years our language will also be easily learnable and researchable by foreign citizens because of the hard work of our linguists and programmers.



**Figure 7**. The social impact of our project to Azerbaijani and foreign citizens.

# Implementation

## Hardware Design

The most valuable for our project is a fast working computer with access to Internet. For all the testing and creating purposes we used a simple computer on Windows. There are no more explicit requirements for the work of the project. To ease and make the process faster we divided the corpus preparation among four people. Furthermore, the application for testing and checking the results and output can also be downloaded and used on any computer with access to Internet.

## Software Design

The software part of the project was completely written on Python language, because of its high capability and comfort in performing machine learning tasks. Moreover, it is widely used in most cases for the POS taggers of other languages and, overall, NLP systems worldwide.

In order to successfully execute a working part of speech tagger, we made sure that each side of the work was carefully done. We outlined all the necessary tools, software and approaches and divided them equally, as follows:

* A good example of Hidden Markov Model, with all the necessary features in it.
* Some basis for the construction of the text corpus
* The vocabularies and dictionaries of Azerbaijani language
* Clear and understandable morphological rules of the language
* The material of proper word tagging.

After discussion about the preferable methods of implementing of POS tagger for Azerbaijan language, we decided to divide the work to two parts:

* Text Preparation
* HMM training and testing

Text preparation has been a very controversial issue, as it was hard to classify the words in Azerbaijani language in some groups. Therefore, after researching about language specialties, we observed that a great amount of words in Azerbaijani always remain the same part of speech, and never change the meaning and there are a lot of these words. These words are planned to be inserted first and taught first to get rid of a big portion of vocabulary. Furthermore, we discovered that each part of speech has its own features, which are inherent to it. For instance, the suffixes or location in the sentence. We considered this as a big plus, as it would be easier to teach the system based on this feature. Azerbaijani language is a language which manipulates words with the help of many different suffixes, with each having its own significance. However, the stem (root) of the word stays the same and often it means only one thing and possess only one part of speech. For example, the stem of the word “silmək” is “sil” which means “to remove” and is a verb. And in this manner, we could eliminate a lot of words, being already tagged for our text corpus. Therefore, we obtained a stemmer program for Azerbaijani language. The principle of this program is to throw out all the suffixes and leave only the stem of the word. The program works by detecting the suffixes by comparing them with a small collection of possible suffixes until the word is in its rooted version. The words are all taken from the input, preprocessed to further be stemmed. The preprocessing of words is very important, because it checks if the word is ready for stemming or not. For instance, all the punctuation being joined to the word has to be taken care of. Therefore, all the punctuation like, for instance, quotation marks or hashtags or brackets are deleted from the words, because they do not possess any morphological meaning. However, all the commas, points, exclamation marks question marks are processed differently. We decided to separate them from the word and give tags to them. After each word is ready, the program starts the stemming process. The accuracy of the program was about 85% which was a good result to proceed. We will be using this in future to stem the words before tagging.

## Essential Components of the Project

The main components are:

* Stemming Program – program which gets a text, preprocess the words and finds the stem of the word.
* Tagging component – a tool which takes the processed stem, compares it with vocabulary and puts the tag after the original word from the text.
* HMM learning tool – a Python component which takes tagged data as input, finds transition and emission probabilities, soothes them and writes to a file with all probabilities “hmmmodel.txt”.
* HMM decoding tool – a Python component which takes untagged text as input, decodes the “hmmmodel.txt” file, performs Viterbi algorithm and writes the tagged text to an output file “hmmoutput.txt”.
* Tags checking program – a program which is used to scan the text and find the tags which are written with errors, either from the vocabulary or when doing it manually.
* Words.txt - list of stems that is used by Stemmer component in order to find the stems of the words.
* Suffix.txt - list of suffixes that is used by Stemmer component in order to cut inflectional suffixes from words in order to find stems.
* Vocabulary excel file – a file from which the tagging component gets the values and tags.

The next step was to start text preparation process. In order to do it, we started developing a vocabulary which will be big enough to have a great portion of words in Azerbaijani language. The most suitable vocabulary for our purposes was Obastan dictionary. We could acquire a vocabulary with about thirty-five thousand tagged words. This was not enough, so we started to work in the direction of broadening this vocabulary by manually adding and tagging words and, eventually, we had a vocabulary with eighty-four thousand of tagged words. Azerbaijani language has 12 parts of speech and we tagged them as:

{is.; f.; sif.; zərf.; say.; əvəz.; qoş.; bağl.; ədat.; modal.; nida.; təqlidi.;}.

In addition to them we distinguished tags for punctuation as follows: for point, exclamation and question marks the tag is “.”, for commas and semicolons the tag is “,”, the dashes and colons remained the tags as they are. This made overall of 16 tags.

After all this work done, a program was developed which would eventually stem the word, compare with our vocabulary, retrieve the matching tag and deliver the original word in a manner of: word/TAG. On the first stages of the development of this program we had problems with timing, because the amount of data in excel file to open each time and search was large and the whole time was calculated as 84000\*(number of words) iterations. We could optimize the algorithm, using hash map method to ease and speed up the access to data, and now it is working at its best, fast as it could be.

The next step was to start collecting data for text corpus. It took a lot of time, because we need a large dataset, properly pre-formatted. Retrieving data from news, books and articles we could manage to obtain 6000 sentences. After formatting this text, we ran it through our rule-based tagger. The results were very satisfying. Those words that were not found in our vocabulary, our tagger tagged as “NOTDEFINED”. The tagger tagged about 70% of the text and we continued to tag the rest manually. It is obvious that some of the tagged words were wrong, but these were errors of the stemming program, which we also handled manually. In addition to that, we wrote another program for the sake of tags accuracy. This tag checking program helped us to detect all the tags which are out of our confirmed format. These mistakes could be the result of some vocabulary-based unclearness, or mechanical error.

After deciding on trigram and bigram mixed HMM model, we acquired the system. It consisted of three parts, organizing and parsing input text, training the model and decoding it with predicting the maximal likelihood. Firstly, when getting the input the programs starts to organize the text and prepare it for learning. With the help of “data.py” the systems detects words and their tags, end and start of sentences and splits the corpus into equal parts for convenience in further processes. The main learning and counting processes are in “hmmlearn.py”. It scans the tags and counts all possible “trigram\_triplets” and “bigram\_tuples”. After this, it sets the transition and emission probabilities. Another very important detail is that it uses the “singleton” counting. These are the tags and words which are final and they appear only once in the corpus. This will play significant role when smoothing emission and transition probabilities. Rule of interpolation and back-off smoothing were used in the project to increase the accuracy. After the smoothing process the sets of transition and emission probabilities are ready to be decoded. All the tags, unique tags, smoothed transmission and emission probabilities, singletons, bigrams and trigrams are written to the output file “hmmmodel.txt” as serialized bytes to save time and memory. The “hmmdecode.py” file is responsible for decoding the “hmmmodel.txt” file and build the most probable set of tags. The decoding is mostly about applying all the gathered data in Viterbi algorithm. After the Viterbi algorithm is done, the program constructs tagged text and passes it to “hmmoutput.txt”. This is our final output file with all of the text being tagged.

## Timeline

Understanding the complexity and seriousness of the project, we constructed our approximate timeline. It was necessary to understand all the project’s features and tools, therefore we spent about 3 months, from October to Jauary, on research of various type. After deciding on the direction of work, we started to look for the necessary tools online and getting to know them. We understood that we will need a huge text corpus, so we started to gather data from various sources. After our Stemmer was ready to work, we started our Data Labeling process. This process was the most time consuming. Starting from mid-March, we began to test our corpus which we had at that time. As the time passes we observed that the performance of the tagger significantly increases, because we continued to work on the tagger. After we reached 20.000 tagged words, we gained the maximal accuracy among all we had.

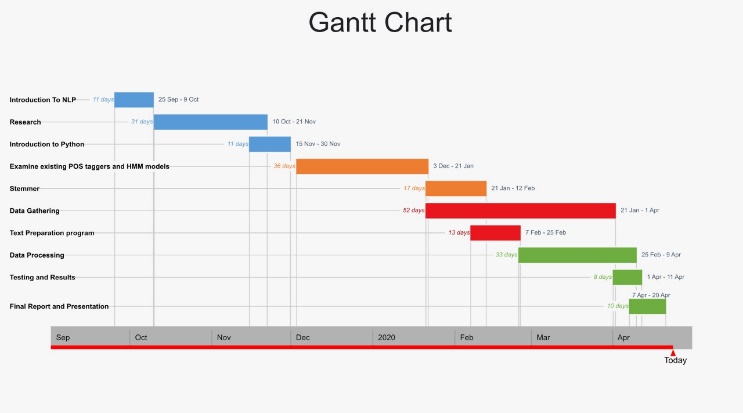


Figure 8. The Gantt Chart

## Testing/Verification/Validation of results

Finally, simple accuracy checker and graphical user interface were created. The accuracy checker works by comparing an original, correctly and manually tagged text with an HMM tagger’s output. The GUI that we built is just a comfortable graphical unit to use and execute all our tools in one place.

Having a small text corpus ready we run it through the HMM model and the results were about 85% accuracy. These results were quite satisfactory for us, as the text corpus has to be much larger. Our investigation shows that, running the machine learning program with the tagged corpus of 2000 words makes the accuracy almost twice lower than with our tagged corpus of 20,000 words. Therefore, it is obvious that increasing the corpus size can considerably increase the systems accuracy. That is why we continue our work on the text corpus. With our rule-based tagger ready, we have eased our manual work significantly, and now the capability for processing larger texts increased sharply.

# Conclusion

### Discussion of results

In conclusion, the PoS tagger that our team implemented showed the capability to substantially assist all the NLP related systems and tools. Firstly, our team used and adapted the stemming program to delete suffixes. Using the tree structured approach, it chooses maximum length stem after comparing all other stems. Then we substantially increased the dictionary and cleared it from all the unnecessary information. After all this done, the dictionary has become a lot more effective and helpful, and we decreased redundancy. Then, a tagging program was written, in order to ease the process of corpus building. This tagger alone can be used to tag some primitive and simple sentences, because the accuracy for this kind of sentences is also high. By building this tagger we could simplify our text corpus work. Some prepositions and punctuation which appear a lot in texts are always tagged correctly and we do not spend any more time on this kind of words. However, for more complex purposes, this tagger cannot show high performance. And, finally, a Hidden Markov Model PoS tagger was obtained and adapted for our purposes. This program calculates transition and emission probabilities to tag the words. It has showed a high accuracy percentage for other languages with big text corpus; however, we could optimize it to 85% for the corpus obtained at the moment. It is believed that, in order to increase the accuracy of this system a much larger corpus will be needed, because the higher the current accuracy is, the harder it is to increase it further.

### Future work

In the future, we are going to build a large corpus because for now we have only 6000 sentences. In addition to this, we can try some new algorithms and approaches to get better results comparing with today.

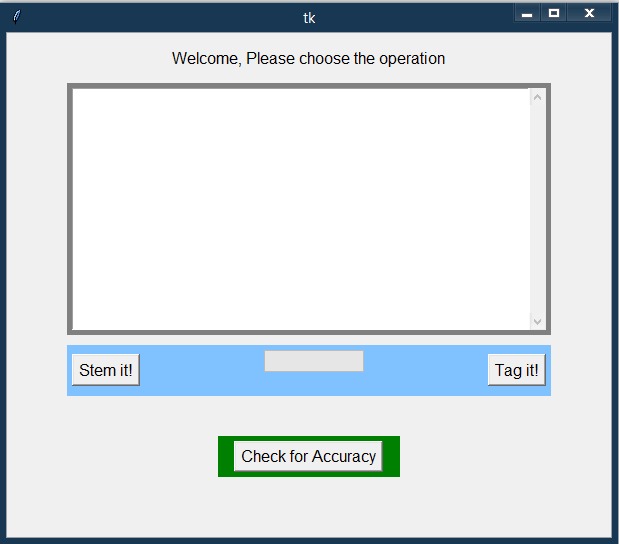
# REFERENCES

1. D. Jurafsky and J. H. Martin, “Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition”, 2008.
2. B. Merialdo, "Tagging English Text with a Probabilistic Model," 1994.
3. T. Brants, "TnT: a statistical part of speech tagger," in Proceedings of Sixth Applied Natural Language Processing Conference ANLP-2000, Seattle, WA, 2000.
4. J. B. Lovins, "Development of a Stemming Algorithm," Mechanical Translation and Computational Linguistics, 1968.
5. B. Can and N. Bolucu, "Joint PoS Tagging and Stemming for Agglutinative Languages," in 8th International Conference on Intelligent Text Processing and Computational Linguistics, 2017.
6. B. Dincer, B. Karaoglan and T. Kishla A Suffix “Based Part-of-Speech Tagger for Turkish,” International Conference on Information Technology: New Generations, 2008.
7. A. Valizada, "Development of mathematical and software applications for PoS tagging texts in Azerbaijani language," 2015.
8. Rustamov Samir, Mustafali Ali, Sadigov Ziyaddin, Mollayev Rasim and Mammadov Samir. “Part-of-Speech Tagging for Azerbaijani Language,” [12th International Conference on Application of Information and Communication Technologies (AICT)](https://ieeexplore.ieee.org/xpl/conhome/8742689/proceeding),2018.
9. Sachin Malhotra, “HMM-Part-of-Speech-Tagger”, GitHub repository,”< https://github.com/edorado93/HMM-Part-of-Speech-Tagger>,”2018.

# 

# Appendices

## Screen shots of the software interface



### List of Figures

|  |  |  |
| --- | --- | --- |
| No | Figure Caption | Page |
| 1 | The Markov chain depicting hidden states (tags) of the HMM as well as transition probabilities between them denoting prior probabilities | 7 |
| 2 | The Markov chain depicting hidden states (tags) of the HMM as well as B observation likelihoods of words. Each tag is associated with a list of probabilities, one for each word | 7 |
| 3 | The working process of Viterbi algorithm. | 8 |
| 4 | The research diagram | 8 |
| 5 | The general architecture of the Stemmer. | 8 |
| 6 | The general architecture of the HMM. | 9 |
| 7 | The Social impact of our project to Azerbaijani and foreign citizens | 11 |
| 8 | The Gantt Chart | 13 |
|  |  |  |

### List of Tables

|  |  |  |
| --- | --- | --- |
| No | Table Caption | Page |
| 1 | Parts of speech in Azerbaijani language and translations | 3 |

### List of Abbreviations

|  |  |
| --- | --- |
| Abbreviation | Explanation |
| NLP | Natural Language Processing |
| PoS | Part-of-speech |
| HMM  RF | Hidden Markov Model  Relative Frequency |
| TnT | Trigram Tagger |
| TBL | Transformation Based Learning |