Development and Design of Deep Learning-based Parts-of-Speech Tagging System for Azerbaijani language

Shafahat Sardarov

Department of Computer Science

Khazar University

Supervisor

Saeed Saeedvand

In partial fulfillment of the requirements for the degree of

Master of Science in Engineering in Computer Science

March 28, 2022

Acknowledgements

Primarily, throughout the course of this investigation, I want to convey my sincere appreciation to my supervisors, Professors Saeed Saeedvand and Behnam Kiani Kalejhi, for exposing me to this research subject, as well as for providing me with invaluable direction and unwavering support. I owe them an enormous debt of gratitude for their consistent guidance and assistance throughout this project's duration and conclusion.

In particular, I want to use this opportunity to thanks every faculty, staff members of Computer Science, the administration of Khazar University, and the Dean's Office for their direct or indirect assistance in a variety of ways during my research project.

Finally, I would want to offer my frankly appreciation and recognition to my parents for their unwavering support, cooperation, and expense during my dissertation and beyond. It would be difficult to mention everyone who has inspired, taught, and helped me. I would want to express my appreciation to all of my well-wishers who have helped with the completion of this thesis, whether directly or indirectly.

Dedication

This dissertation committed to all of family members who helped me a lot while I trying to complete my thesis.

Abstraction

Parts-of-Speech (POS) tagging, also referred to as word-class disambiguation, is one of the prerequisite techniques that are used as part of the advanced pre-processing stage across pipeline at the majority of natural language processing (NLP) applications. By using this tool as a preliminary step, most NLP software, such as Chat Bots, Translating Engines, Voice Recognitions, etc., assigns a prior part of speech to each word in the given data in order to identify or distinguish the grammatical category, so they can easily decipher the meaning of the word.

This thesis addresses the novel approach to the issue related to the clarification of word context for the Azerbaijani language by using a deep learning-based automatic speech tagger on a clean (manually annotated) dataset. Azerbaijani is a member of the Turkish family and an agglutinative language. In contrast to other languages, recent research studies of speech taggers for the Azerbaijani language were unable to deliver efficient state of the art accuracy. Thus, in this thesis, study is being conducted to investigate how deep learning strategies such as simple recurrent neural networks (RNN), long short-term memory (LSTM), bi-directional long short-term memory (Bi-LSTM), and gated recurrent unit (GRU) might be used to enhance the POS tagging capabilities of the Azerbaijani language.

Since we do not have a well-structured open-source annotated corpora for our language and the datasets used for other taggers are not publicly available, it would be difficult to compare our results with previously developed taggers. As a result of this, during the initial part of the study, a native Azerbaijani speaker collected a range of blog posts and documents that were written in Azerbaijani, revised them, and tagged them with the proper categories. Afterwards, a number of different machine learning approaches from a few recent studies, including the HMM & Viterbi with Stemmer, and Conditional Random Fields (CRF), were used as a baseline to train a clean annotated corpus for Azerbaijani.

In the second stage, all approaches were measured based on accuracy, f1-score, precision, and recall by using corpora that we built for the Azerbaijani language, which comprises approximately 20k words. From stochastic machine learning models, whereas HMM-Stemmer achieved an accuracy score of approximately 87%, and CRF has 88% score. The highest rounded scores for RNN, GRU, and LTSM, respectively, were 88%, 92%, and 96%, and Bi-LTSM outperformed other models with a 98% accuracy score.

Our suggested solution implements RNN, GRU, LSTM, Bi-LSTM approaches on the manually tagged Azerbaijani corpus. By using fastText we transform data, and it passes through deep neural network which uses all abovementioned deep learning algorithms as hidden layers to perform parts of speech tagging.

Referat

Nitq hissələrinin etiketlənməsi (POS) prosesi, həmçinin sözün leksik sinifinin qeyri-müəyyənliyi ilə də tanınır, təbii dilin emalı ilə bağlı olan əksər tətbiqlərin yaradılması prosesində tələb olunan üstün ön emal mexanizmlərindən biridir. Söhbət botları, tərcümə motorları, səs tanıma sistemləri və s. kimi bir çox təbii dilin emalı programlarında, bu alətdən ilkin addım kimi istifadə etməklə, sözün qrammatik kateqoriyasını müəyyən etmək və ya ayırd etmək ücün verilənlərdə olan hər sözə ona uyğun nitq hissəsi etiketi əlavə edilir və beləliklə, bu programlar sözün mənasını asanlıqla qavraya bilirlər.

Bu tezisdə təmizlənmiş (manual olaraq etiketlənmiş) verilənlər setində dərin öyrənmə əsaslı Azərbaycan dilidə nitq hissəsinin etiketlətməsinin avtomatlaşdırılmış prosesdən istifadə etmək ilə söz kontekstinin aydınlaşdırılması ilə bağlı məsələyə yeni yanaşmadan bəhs edir. Azərbaycan dili iltisaqi dildir və Türk dil ailəsinə daxildir. Digər dillərdən fərqli olaraq, Azərbaycan dili üçün nitq hissələrinin etiketlənməsi prosesinin rəqəmsallaşdırılmağı ilə bağlı son tədqiqatlar effektiv deyildi və digər dillər üçün olan müasir araşdırmalarda əldə edilən dəqiqliyi təmin edə bilmədi. Məhz bu səbəbdən, bu tezis sadə təkrarlanan neyron şəbəkələri (RNN), qapılı təkrarlanan mexanizm (GRU), uzun qısa müddətli yaddaş (LSTM), iki tərəfli uzun qısa müddətli yaddaş (Bi-LSTM) kimi ardıcıl dərin öyrənmə algorithmlərindən istifadə etməklə Azərbaycan dilində nitq hissəsi etiketlənməsi prosesinin rəqəmsallaşdırılmasını necə inkişaf etdirə bilirik sualının cavabını araşdırır.

Dilimiz üçün yaxşı strukturlaşdırılmış açıq mənbəli etiketlənmiş data korpusumuz olmadığından və digər avtomatik etiketləyicilər üçün istifadə edilən verilənlər setinin ictimaiyyətə açıq olmadığından, nəticələrimizi əvvəllər işlənib hazırlanmış programlarla müqayisə etmək çətin olardı. Buna görə də, tədqiqatın ilkin hissəsində doğma dili Azərbaycan dilində olan şəxslər Azərbaycan dilində yazılmış bir sıra bloq yazıları və sənədləri toplayıb, onları yenidən gözdən keçirib, və müvafiq nitq hissələri ilə işarələyib. Daha sonra, Azərbaycan dilində olan bu korpusu əvvəlki tədqiqatlarda istifadə olunan gizli markov modeli (HMM) və şərti təsadüfi sahələr (CRF) kimi məhşur maşın öyrənmə alqoritmlərindən istifadə edilərək öyrədilmiş və nəticələr əldə edilmişdir.

İkinci mərhələdə yuxarıda sadaladığımız bütün dərin öyrənmə alqoritmlərin dəqiqliyi təqribən 20 min sözdən ibarət Azərbaycan dili üçün qurduğumuz korpusdan istifadə etməklə müəyyən metriklər əsasında ölçüldü. Stoxastik maşın öyrənmə modellərindən, HMM təxminən 87%, CRF isə 88% dəqiqlik balına malikdir. RNN, GRU və LTSM üçün ən yüksək

yuvarlaqlaşdırılmış ballar müvafiq olaraq 88%, 92% və 96% olub və Bi-LTSM 98% dəqiqlik balı ilə digər modelləri üstələyib.

Bizim təklif etdiyimiz yanaşma RNN, GRU, LSTM, Bi-LSTM algoritmlərini bizim tərəfimizdən Azərbaycan dili üçün yığılmış etiketlənmiş verilənlər setinin üzərində tətbiq edir. FastText-dən istifadə etməklə biz etiketlənmiş məlumatları vektor formatına çeviririk və onlar nitq hissələrini etikətləmək üçün yuxarıda qeyd olunan bütün dərin öyrənmə alqoritmlərindən gizli qatlar kimi istifadə edən dərin neyron şəbəkəsindən keçir.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Figuring out how to give machines the ability to grasp things that are written or said in human language has proven to be one of the most challenging and time-consuming difficulties facing academics in the discipline of artificial intelligence (AI). If we think about how lazy the nature of human beings is compared to computers, the desire that never stops to facilitate their job and to fulfill the need to interact with the machine in a language more similar to that used in everyday conversation, gives birth to natural language processing (NLP). NLP is the branch of computer linguistics that deals with the challenges of communication between human beings and machines computationally. The majority of NLP applications we use in daily life, from text-to-speech conservation to the question-answering add-ons seen on websites, need underlaying technologies before the actual software is built for the purpose of successfully completing a variety of activities and jobs like parts-of-speech (POS) labeling, information retrieval (IR), and named entity recognition (NER).

Processing natural language is now considered one of the most promising study fields by both academic and professional experts. The ability to parse and comprehend language is the main objective, but this aim has not yet been completely met. Therefore, NLP research has concentrated on preprocessing and intermediate tasks, which make sense of intrinsic language structures without needing a full comprehension of the structures themselves.

One of these activities is known as POS tagging, which is used to annotate words with prior labels, which helps machines understand the grammatical class of words such as nouns, verbs, and adjectives, so they can later differentiate and find true representations for a given context. For wide-spread languages like English, Dutch, and Chinese, speech taggers have already reached the desirable level of accuracy, scoring 97%, which is equal to highly educated professionals. However, low-resource agglutinative languages, like Azerbaijani languages, it is still considered as painful experience since we do not have accurate POS taggers that generate enough results in order to compete with others. Beside the difficult nature of the language, the absence of massive, annotated corpora has been the obstacle in the way of designing efficient de-facto POS taggers.

This dissertation considers conducting contribution to NLP oriented studies that has been completed by academics, on the favor of Azerbaijani language to achieve an elevated level of accuracy by adopting deep learning approaches. With the intention of staying loyal to the purpose of this research, tagged corpora and proposed solution will be open-source, so everyone can access and contribute to solution.

1.1 Problem Statement

The Azerbaijani language, a descendant of the Turkish language family, is also known as Azari or Azerbaijani Turkic, and is the most commonly spoken by Azerbaijanians in our domestic country, Azerbaijan. On the other hand, it is employed for communication by a significant population in the northwestern part of Iran, in certain regions of Georgia and Turkey, as well as by communities all over the globe, especially in Russia. It is the official language of Azerbaijan, and along with that, it is among the declared official languages in the Russian federal province of Dagestan. Statistics shows that there are more than 22 million Azerbaijani speakers.

As it has been mentioned before, the process of generating a list of words and associated part-of-speech tags is just one aspect of POS tagging. It can be easily observed that every language contains a substantial number of terms that may indicate multiple part of speech depending on the context at the same time. This rule of thumb is valid for the Azerbaijani language too.

Azerbaijani is one of the intense languages in terms of morphology and it composed of overall eleven parts of speech with two distinct groups (Table 1) as six main and five auxiliaries. However, there are a couple of language-related challenges we can come across during the tagging process.

Initially, all languages, including Azerbaijani, often run into issues with homonyms, which are words that may be used in more than one manner, contingent upon situation, throughout the disambiguation procedure. Perhaps, "kök" in the phrase "kök adam" – "fat man" is an adjective, whereas "kök yedim" – "I ate carrot" is a noun. This could lead to potential trouble for the tagger.

Additionally, if we look at the following sentence, "Sizin almaq istədiyiniz maşının mülkiyyət hüququ bu cavana aiddir." – "The ownership of the car you want to buy belongs to this young man." In this sentence, "cavan" and "young" are adjectives and not homonyms but function as nouns, which is called "substantive adjective." Due to this challenge, the application cannot

automatically mark the word with the part of speech annotation that matches its intended category.

№	Parts of speech	Azerbaijani / English
1	Isim=Noun	Çay=Tea
2	Sifət=Adjective	Təhsilli=Educated
3	Say=Numeral	Doqquz=Nine
4	Fel=Verb	Fikirləşdi=Thought
5	Zərf=Adverb	Bütün=Whole
6	Əvəzlik=Pronoun	Bu=This
7	Ədat=Particle (grammatical)	Yəni=I mean
8	Modal=Modal	Beləliklə=So
9	Bağlayıcı=Conjunctive	Hərçənd=Though
10	Nida=Interjection	Vay=Ouch
11	Qoşma=Postposition	Sarı=Towards

Table 1. Azerbaijani Parts-of-Speech (Nitq hissələri)

Moreover, Azerbaijani is not only restricted to eleven lexical word categories, but there are also some kinds of words that do not particularly belong to any part of speech that asks for a new identifier. The particles "idi, imiş, var, yox" which equivalent to respectively "was, were, there is, there is no" in English, are the samples of such sort of words. Likewise, various words that are nouns themselves but, after combination with numerals, behave as "numerative" words. Few examples of numerative words are the followings: "damcı" as noun translates as "drop," "iki damcı su" is "two drops of water," "5 ton" same for both languages and so on.

The fourth concern of Azerbaijani is that every agglutinative language is also a highly inflective language, which means that words may be interpreted in a number of diverse ways owing to the suffixes and prefixes that they include. To put it another way, the several types of suffixes

that may be added to the end of a root word are not subject to any restrictions of any kind. It is possible to see that a single word in Azerbaijani requires a lot of translations into English. According to (Fatullayev, 2008) we can generate in excess of 8000 distinct types from one stem. Let's look at following example, the word "uzaqlaşdırılmışlardansınızmı" which may be interpreted as "Are you one of those who have been expelled?" is composed of nine suffixes added to the root word "uzaq" corresponding to "far" in English.

Next issue we face while trying to disambiguate the words is about stressing. "Alma alma" is translating as "apple apple" if we use Google translator, but it should be translated as "Don't buy apple." When we speak, we stressed second syllable it means root of word is apple and grammatical category is noun, for second word of sentence we stress first syllable it is because root of word is "al" which means buy and it should be tagged as verb.

Furthermore, the Azerbaijani language has some forms of verbs that may operate as nouns, adjectives, and adverbs analogous to English, but in contrast to English, they should be classified as in lexical terms:

"Burada torpağı qazan adamı tutdular." – "They caught the man digging the ground here."

The word "qazan" is "feli sifət" – "participle" in the above example.

Considering all of the mentioned problems, we can say without any doubt that Azerbaijani's morphological structure makes categorizing parts of speech a challenging assignment. Solving issues that are produced by the characteristics of the Azerbaijani language will be quite a laborious and complex mission. This is the first reason to motivate us do research in this field.

Another reason, we do not have a dependable and publicly available data corpus for our language. It doubles the trouble for a solution in turn. It is obvious that without whenever we try to modify and optimize taggers if we do not have same corpus, it will cost our accuracy since parameters obtained from one sphere to another would not be constant. In spite of fact that possessing trained corpora is essential for improvements in the NLP technologies, there have been no real efforts made corpora for taggers. There is also big gap related to scarcity of big corpora that have been manually tagged for the sake of training.

Due to extensive increasing number of users who using the internet, the amounts of data have been consuming is becoming to grow exponentially, researchers in NLP field are starting to tend data-driven approaches, because it can help to build efficient models (Wu, et al., 2020). Similarly, upsurge in computing power like improvements in GPU, even most recently

introduced by Google TPU (Cass, 2019), open the doors to usage of deep learning algorithms in NLP. Novel practices achieved to remarkable metrics in terms of accuracy, so it yields radical shift from old methodologies. It makes development simpler compared to traditional stochastic and rule-based concepts. It also applicable that since sequence in human language is important, RNN models are also sequential models, we are able to get highest metrics even small set of data.

Last but not least matter motivates us, there are two POS tagging system for Azerbaijani language and both of them are using traditional Hidden Markov Model (HMM) as core algorithm for their applications (Valizada, 2015; Mammadov, et al., 2018). Both of them is having accuracy points which is lower than 90%. One of the few language technology contributors from Azerbaijan was an organization called "Dilmanc." It was a project under the of the Ministry of Digital Development and Transport of the Republic of Azerbaijan. As a part of the project, various applications have been created and shown at global exhibitions such as voice recognition, machine translation, and a text-to-speech system of varying degrees of quality are among them (Mammadov, et al., 2018). Conversely, corpus they have been employed, not yet available to the general public, and the project does not use open-source software and also by the time now it is suspended (Mammadov, et al., 2018). Taking into account that it is important to have a speech tagger in the majority of NLP applications, there should be implemented novel DNN-based POS taggers with high precision for Azerbaijani language and also annotated datasets that serve this purpose.

1.2 Objectives and Contributions

1.2.1 Objectives

The fundamental aim of the thesis is to design a POS speech labeler for Azerbaijani language that has a satisfactory degree of accuracy. In order to achieve this wide-range purpose, we have established the sets of goals:

- Researching multiple machine learning and deep learning algorithms that are accessible is a necessary step if we are going to develop a POS tagger for the Azerbaijani language.
- The morphological possibilities in Azerbaijani are almost endless. To be able to construct a POS tagger with limited the resources that we have available to us, we would want to make use of the morphological characteristics of a word, as well as the word's suffix.

- Because there was no Azerbaijani corpus that we could access, we were pushed to start developing resources for the language. The process of manually identifying parts of speech is one that is both time consuming and challenging. As a result, we want to focus on developing approaches that will allow us to carry out the process of tagging parts of speech successfully while using a minimal number of labeled resources.
- The study also involves the establishment of a fairly good quantity of annotated corpus for the Azerbaijani language, which will directly assist the implementation of a number of different NLP software.
- In conclusion, we intend to perform a series of experiments with the aim of investigating the applicability of various machine learning approaches. In addition, we will conduct comparison research of the precisions produced by operating with various POS annotating techniques.

1.2.2 Contributions

The following is a synopsis of the most important contributions made by this thesis:

- The POS tagging procedure, which can be thought as a preliminary stage for a lot of software and systems dealing with natural language processing, is explained in detail.
- Explores the challenges that arise when attempting to tag speech in languages with limited resources, such as Azerbaijani, and provides an inventory of the most cutting-edge methods for doing so.
- Azerbaijani, which belongs to the Turkish language family, is the focus of this research. Indeed, we want to call attention to the fact that as a direct outcome of the effort, a resource that is made up of 20,000 different POS tagged corpus has been produced.
- The primary purpose of the work is to implement four deep learning strategies into the process of POS tagging in Azerbaijani. RNN, GRU, LTSM, and Bi-LTSM are the four commonly used deep learning approaches that we have utilized in order to solve the Azerbaijani POS tagging. We have obtained extremely basic models based on the aforementioned deep learning approaches, and our testing of the acquired models has shown that they are sufficient.
- The target of this dissertation is to deliver a description of the prototype model, carry out an exhaustive analysis of the strategy that was presented, and carry out experimental

verification of the performance. In addition to that, this thesis performed experimental measurements of the efficiency.

1.3 Overview of the Thesis

- In Chapter 1, the study is described, which serves as an introduction. This contains a short explanation of the reasons behind our motivation of the study, the thesis objectives and contributions to field, the relevance of this research, the purpose and goals of the research, as well as the scope and constraints.
- In Chapter 2, The review of the literature and works from previous examinations may be found.
- In Chapter 3, contains a discussion of all the models' specifics on their implementation.
- In Chapter 4, we go into the methodology that was used in this study. It includes the design of the baseline algorithm as well as the architecture of deep learning network.
- In Chapter 5, the assessment and findings of the models constructed throughout the course of the study are shown,
- In Chapter 6, ultimate conclusion of this research effort is presented.

Chapter 2

Literature Review

Summary

The object of this chapter is to provide a broad outline of the current related trends in the primary fields connected to the POS tagging system discussed in the thesis. Surveyed research contains mostly solutions for agglutinative languages. POS taggers have four main directions to examine the following subjects:

- Rule-based approaches
- Stochastic or Probabilistic methodologies
- Hybrid or Transformational taggers
- Deep learning models

We do not want to provide an exhaustive evaluation of the work since we want to preserve reader's interest. Instead, we will provide a concise overview of the many methods that were used.

2.1 Parts-of-speech tagging approaches

The contributions made by a number of academicians over the course of the previous few decades have been responsible for the tremendous expansion that has occurred in the discipline of autonomous Part-of-speech tagging. Many innovative ideas have been presented with the focus on boosting the capability of the tagger and to create the POS taggers for a number of different languages from its debut in the middle sixties. In the very beginning, researchers manually developed the rules for the tagging process. Speech taggers are programmed with a set of rules or restrictions that were developed by linguists and integrate this knowledge. Then new era has start with a number of statistical and probabilistic models also referred to as stochastic approaches have been employed for the challenge of generating versatile and customizable POS taggers. Numerous complex algorithms for machine learning have been created, and these algorithms collect more reliable data. When it comes to learning the core model, every stochastic methodology, on the whole, depend on manually annotated dataset. This is a challenging skill to master for a new language. In light of this, dozens of popular studies have concentrated their attention on unsupervised and semi-supervised learning-based

methodologies in order to overcome the lack of source issues. Current trends are using deep learning algorithms for establish tagging systems, and it could be said that they have already beat old-school probabilistic models and rule-based approaches in terms of metrics.

2.1.1 Rule-based approaches

A two-step design was employed to automatically allocate part of speech annotations in the early stages of development of POS taggers. At the start, potential tags were identified for each word by consulting a dictionary if word was having one parts of speech it was jumping to next word. In case of words have more than one annotation, during the second step of the process, an enormous number of hand-written classification rules were applied in order to narrow down this record to a prior part of speech. Although the vocabularies and set of rules used in modern rule-based techniques to label parts of speech are far bigger than those used in the sixties, the architecture of these systems is quite similar.

If the word before the one in question is an article, for instance, the word in question must be a noun. The information is encoded using these principles as a coding system. In finite state automata, context pattern principles combined with lexically ambiguous sentence representations could be used (Brill, Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging, 1995). Alternatively, the rules could be regular expressions.

Rule-based taggers need a significant amount of linguistic expertise to establish their rules since they employ a set of manually constructed rules. Because of this, the process of manually defining rules takes a significant amount of time and requires significant work from humans. It creates a challenging environment for the development of rule-based taggers.

Every single word has been assigned a one-of-a-kind code that is determined by the lexical category that it falls within. The tag set is used to determine the part of speech labels that match to the terns that are included inside of the dictionary. The processing of the tagger begins with the tagger checking up each word in the vocabulary to determine which segment of speech corresponds to each word. Tag altering rules are the rules that offer information about whether or not a particular tag is acceptable given the context in which it is being used. Such guidelines may state that a noun must come after a determiner and an adjective in a sentence in order for the sentence to make sense. Rules have two types it can be either contextual rule, which are established rules depending on the context, or lexical rules, which assist the application in

making fair guesses. Contextual rules are the most common kind of rule. The label of the word is changed by contextual rules depending on the words that are around it, while lexical rules make use of the grammatical nature of the word directly. The majority of rule-based methods make use of contextual data in order to attach tags to words whose meanings are misleading.

$$determinant - X - noun = \frac{X}{Adjective}$$

From the formula we may state something along the lines of "identify it as an adjective, if an unidentified word X is accompanied by a determinant and preceded by a noun."

One of the objections against rule-based taggers is the amount of labor necessary to construct the disambiguation rules. A large amount of effort is necessary when utilizing a rule-based tagger to construct a rule set. Furthermore, the rules utilized in rule-based systems are usually difficult to build and, in most circumstances, do not display resilience. Even in a broader context of syntax, it is not difficult to develop a clear set of logical norms, and this is something that is necessary the majority of the time. However, tuning such systems to achieve high performance is a time-consuming and labor-intensive procedure.

2.1.2 Stochastic Methodologies

Machine learning models for tagging part of speech is still considered as popular and accurate choice for most of languages. The statistical tagging methods are based on data-driven techniques, such as the automated extraction of frequency-based information from annotated dataset for the purpose of applying it to newly discovered words. The term "stochastic model" may be used to any model that takes into account either frequency or probability. The stochastic taggers separate meanings from words by bearing in mind the possibility that each term will be associated with a certain tag. Approaches like the Conditional Random Fields (CRF), and Hidden Markov Model (HMM) are examples of some of the stochastic methods. These probabilistic taggers provide higher levels of accuracy than their rule-based ones.

Stochastic taggers are only effective on the domain of the dataset on which it was trained because they are using information about probabilities of the trained corpora. In other words, if the tagger is trained on a corpus that contains bio-medical articles and then used to annotate a corpus that contains blogs about history, the tagger will not provide satisfactory results for the new corpus. In addition, the training requires a substantial amount of annotated data to be used.

2.1.3 Transformation Based taggers

Transformation based tagging combines the benefits of both rule based and probabilistic approach. It starts by selecting the tag that the training corpus indicates is the most probable candidate, and next it submits a predetermined collection of criteria to determine if annotation has to be changed to something different. Taking this method will result in a number of mistakes. A further action is to perform the transformation rules that the tagger has learnt in order to rectify as many mistakes as is practically feasible. It makes a note of any new rules that it comes across throughout the process and stores them away for future. The Brill Tagger is a good illustration of the kind of efficient tagger that falls under this category.

One of the strengths of this method can be considered as taking use of a greater variety of syntactic, and lexical patterns that are repeatably. Especially, labels are able to be predicated on words as well as other contexts. In order for us to get a deeper comprehension of TBL, we may make a parallel between the procedure and the act of creating sculpt. Suppose a stone sculptor acting on a lion figure by chipping away at the stone. At initially, he had a cuboid-shaped piece of alabaster in his possession. At this point, he begins to remove large amounts of alabaster in an effort to make the figure resemble the head of a lion. However, he does this without highlighting facial features, adding mane, or anything else of the kind. Next, he works on the model some more, focusing on the finer aspects; he removes a great number of the model's smaller components in order to get a respectable lion resemblance. In the long run, the artist will use a minor blade, also work on small-scale areas, such as the eyes, nose, ears, and mane among other things. This will be done in stages. After a lot of hard effort, the stone artisan is finally able to create a perfect figure of a lion, complete with the desired highlights and elaborated down to the minimum features. Like the artist, TBL functions in an analogous way. In the beginning, TBL will tag the content by selecting the rule from the presented set of labeling guidelines that is the most inclusive one possible. In this approach, it looks for a standard that is more specific, with the goal of producing tags that are just slightly different from those produced by the prior, most general concept. The process will continue until the material is labeled with an adequate level of precision.

By choosing and series of conversions that turn a preliminary flawed labeling into one with less errors, TBL tagging conveys the intricate interconnections that exist between words and tags (Jurish, 2003). On the other hand, this is accomplished by selecting conversions that alter a preliminary defective annotating into one with scarer faults. Estimating the vast majority of

Markov model parameters needs orders of magnitude more choices than training a tagger that is based on transformations, which requires orders of magnitude less decisions. Transformational learning, also known as TBL, often begins with a straightforward approach to the issue at hand. Following that, it repeats in cycles. In each process iteration, the target problem is exposed to the change that will provide the greatest overall benefit. The algorithm comes to a halt when it determines that the changes that have been chosen cannot provide any further value or when there are no more changes that can be chosen.

This would be analogous to drawing a guy on horse, you first need to draw a horse, then draw a man and then painting a man with different color so you can get final accurate result. TBL is most effective when used to categorization problems. Accuracy is often regarded to be the goal function while working with TBL. Therefore, throughout each cycle of training, the tagger seeks for the modifications that significantly cut down on the number of mistakes in the preparation list. After that, this transformation is inserted to the collection of available transformations, and it is used on the training slice of dataset. At the completion of this phase, the program is executed by foremost annotating the raw data with the starting state tagger, then applying each conversion in sequence where it may be used (Getachew, 2001). This process is repeated until the tagger is run successfully.

Eric Brill (Brill, 1995) suggested an additional method to POS tagger that he termed the Brill tagger. It is a kind of transformation-based learning that bears the name of its creator. He suggested using a program that would first make an educated estimate as to the tag that should be associated with each phrase, then go back and correct any errors. The purpose of the tagger is, in general, to allocate each term in a given data the tag that is most likely to apply to it based on an initial-state tagger's estimation, which is trained on a large, tagged corpus without taking context into account. When the text had finished going through the first state tagger, it proceeded to compare itself with the reference text (manually tagged text). As a consequence of this, an ordered list of transformation rules is acquired, which may be used to modify the output of the initial-state tagger so that it more closely resembles the text that was used as a reference.

2.1.4 Deep Learning Models

Nevertheless, Stochastic approaches was performing better with regard to POS tagger, deep learning models started to replace, and even outperform them when it comes to identification part of speech. Bi-directional RNN-based taggers (such as Bi-LSTM) accomplish the label

disambiguation for the entire sentence as a sole choice issue and supply the chance to utilize data arriving from both sides right and left simultaneously. This makes it possible to use information coming from both sides simultaneously. Despite these gains, however, there is a cost associated with training and evaluating a deep learning architecture in terms of computing resources (Perez-Ortiz & Forcada, 2001). In the related works section, we will examine common sequential models that have been applied to create POS tagging systems such as DNN, CNN, MPL, RNN, GRU, FNN, LTSM, Bi-LTSM.

2.2 Related Works

2.2.1 Related works on Azerbaijani language

According to (Valizada, 2015) the Rule-based technique is the one that is the most useful for the study of the Azerbaijani language since, in his opinion, it is the one that makes the best use of both time and resources. In addition to this, he is of the belief that we will be able to get better outcomes with more constrained corpora if we cut down on the number of rules that control stemming and tagging. His position is that a strategy that is based on rules has the ability to achieve an accuracy of around 95%.

The participants of in (Mammadov, et al., 2018) examines the results of employing HMM and Viterbi algorithms in combination with the stemmer for parts of speech tagging using an annotated dataset for Azerbaijani language. The authors perform three stages to tag the words in each phrase using the provided dataset. A stemmer has been developed as a first step, with the purpose of eliminating suffixes from the ends of words in a given text in order to return the words to their root forms. As a result of this, a variety of techniques have been tried. After examining the relative merits of each of these strategies, they found that patterns like tree de-suffixing was the very successful overall. The tree method considers all of the potential stems of a word before selecting the longest available option. Next, dictionary has to be examined to see if any of words are redundant or whether any are absent. After all, program generates the 13x13 transition probability matrix which correspond the number of taggers they have included tag-set based on Viterbi algorithm corpus. The transition matrix as well as a new test data are sent to the tagger. The tagger then marks tags to the words in the test dataset based on the information that is provided in the matrix. This research was conducted using a corpus of data including only 3000 words. They achieved approximately 90% accuracy score for tagger.

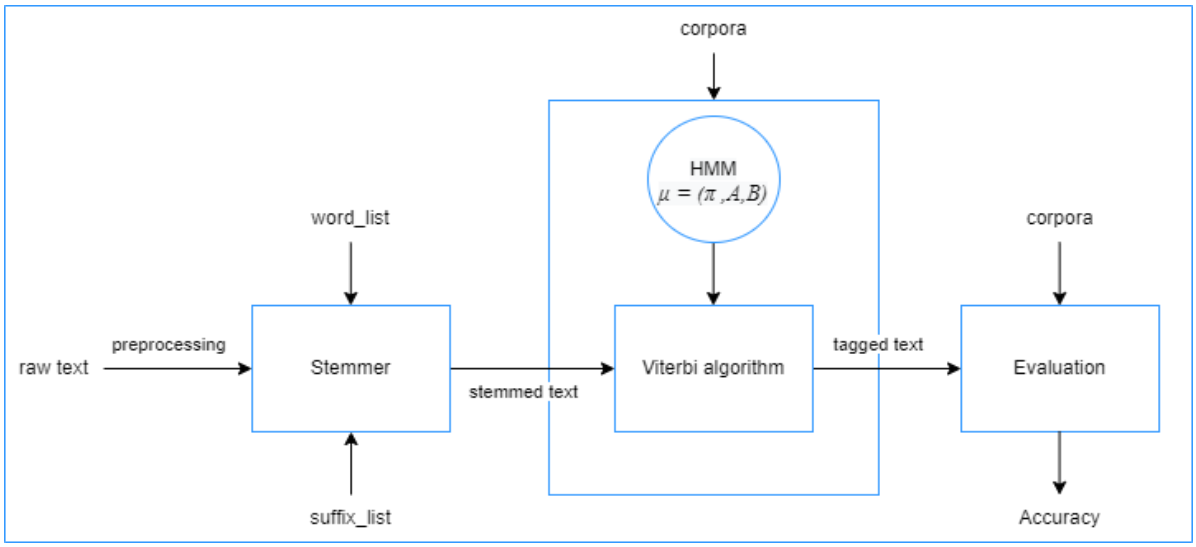

Figure 1. Workflow of HMM-based POS tagger for Azerbaijani

(Mammadov, et al., 2018) states that as a consequence of the requirement of producing stems aimed at newly unlabeled text, the identical stemmer component that is used in the simple PoS annotator application is utilized in the HMM annotator. Figure 1 illustrates work-flow and general design of previous tagger for Azerbaijani language. Following the removal of inflectional suffixes and the completion of the required stages in stemming, the newly formed words are then prepared to be scan, recognized, and tagged by an HMM tagger. The main program of HMM sends fresh untagged stems to the tagger, which in turn sends it a pre-calculated bigram matrix. The tagger then tags new words in accordance with the values of the matrix. Following the labeling process, the annotator will send the words to the main program. The output file is updated with newly tagged content whenever the tagger's main program is run. Main critics about that research, it is tested with small size corpus with 3000 words which is not good criteria to evaluate accuracy.

2.2.2 Related works on foreign languages

2.2.2.1 Rule-based

A rule-based approach is used in the research that was carried out by (Hakkani-Tür, Oflazer, & Tür, 2002) in order to build a morphological tagger for the Turkish language. He demonstrates the limits of the derivational structure of the Turkish language. The tagger is implemented with the help of finite-state automata (FSA), which may be used to collect statistics and fine-tune the morphological analyzer by logging erroneous parses, frequently used roots, and other such examples. According to early statistics, the tagger is capable of correctly tagging approximately

98–99% of texts with just little involvement from humans. Oflazer states (2002) When dealing with morphologically disambiguated texts, LFG parsers cut disambiguation time in half and complete the process 2.5 times faster. During the course of this specific piece of research, they did not employ any automation machine learning methodologies in order to evaluate the tagger.

Another POS tagger that was developed by (Altunyurt, Orhan, & Gungor, 2007) with the intention of classifying Turkish literary works use a composite approach as its foundation. This method makes use of some heuristic components of the language, in addition to combining rule-based and statistical techniques. Additionally, it integrates certain aspects of the language in terms of heuristics. This approach also makes use of the frequencies of individual words as well as the probabilities of n-grams, which are combinations of unigrams, bigrams, and trigrams. With the intention of succeeding a higher level of precision with the system, the findings of a morphological analyzer are combined with those obtained through stochastic methods. The tagger was able to reach an accuracy level equivalent to 80 % its target. This level of accuracy is not enough to race with other POS taggers.

2.2.2.2 Stochastic

(Can & Bölücü, 2019) offer a Bayesian method backed by HMM, which is considering as unsupervised model, is as a solution to lessen the sparsity of stemmed words. This issue of sparsity in POS labeling is tied to the characteristics of agglutinative languages. The problem of using words that are not in one's dictionary arises due to the intricate morphological construction of terms, which consists of both derivational and inflectional suffixes. Because of this, the terms can't be located in the lexicon and hence can't be labeled, which leads to the sparsity problem in the dataset. The participants use the approach outlined to three distinct languages: Turkish, Finnish, and English. They do so by employing of a mixture of stems and suffixes that, when used together, offer outcomes that are superior to those obtained by making use of words that have the same model. When using in conjunction with one another, it is believed that more precise results can be obtained from POS annotating, and the stemming phases.

In the (Kurfalı, Üstün, & Can, 2016) presents another strategy for dealing with the sparsity issue that can be implemented. This method employs Conditional Random Fields (CRF) to produce morpheme tags, which are then processed using HMM. According to the findings, the use of CRF in conjunction with HMM generally has a beneficial impact on sparsity.

To address the problem of incorrect POS tagging, the author of the study (HIRPSSA & Lehal, 2020) proposed an algorithmic prediction of the POS annotations which ought to be applied to terms in the Amharic language. This was done with the purpose of finding a solution to the issue. A comparison is presented between the three POS taggers that are based on statistical analysis. The effectiveness of each of these taggers, which comprise taggers for HMM, Naive Bays, and Trigrams'n'Tags (TnT), was analyzed by applying the same datasets to the training and testing phases of the experiment. The empirical data suggest that the efficiency of the CRF-based tagger is more efficient compared to the taggers that are in direct competition with it. In the course of the test, the CRF-based tagger was successful in achieving the greatest level of accuracy possible, which was 94 %. When compared to previously developed POS taggers which employs CRF algorithm to train data, their performance is not much enhanced. The quantity and kind of feature set are insufficient to increase the tagger's performance.

2.2.2.3 Deep learning

(C. D. Santos & Zadrozny, 2014) have developed a Deep Neural Network (DNN) (CharWNN) that can learn the representation of words at the character level and combine it with the representation of words at the usual word level in order to conduct Part-of-Speech (POS) Tagging. Using this CharWNN, the researchers have also developed two POS taggers, one for the English language and one for the Portuguese language. They have circumvented the need for manually constructed features by adding a layer of convolutional neural networks (CNN), which facilitates the efficient extraction of features from texts of varying lengths.

The work that (Collobert, et al., 2011) was expanded upon by the neural network that was suggested, and it does so by adding a convolutional layer, which learns character level embedding of words. They have used the Collobert's Window technique to score each word in the sentence by using the embedding that mixes word level and character level embedding. Additionally, the Viterbi algorithm has been used in order to determine the unique tags for each individual word. They have completed the unsupervised pretraining of word embedding, which contributes to an increase in the precision of POS tagging. The pre-training was conducted in both English and Portuguese by using the word2vec program with the identical set of settings across both languages. They conducted their evaluates on the Penn Tree-Bank Wall Street Journal corpus, and their results showed that the model had an accuracy of 97.32 % for the English language. Additionally for the Portuguese language on the Mac-Morpho corpus, where it attained an accuracy of 97.47 %.

In the study (Wang, Qian, Soong, He, & Zhao, 2016) a bi-LSTM RNN is developed for POS tagging that included word embeddings, and they achieved a level of tagging accuracy that is considered to be state of the art. In addition to this, they have shown an original training strategy for word embedding. Suggested model was built by using the machine learning library called "CURRENT" in its construction. With the aim of training the word embedding, they used around 536 million words from unlabeled news articles from North America. They have evaluated the proposed model with a number of different sizes for the hidden layer in order to determine which one is the most effective. The proposed model was assessed with the use of data from Penn Treebank sourced from the Wall Street Journal. For the suggested model that was recommended, an accuracy of 97 % was achieved.

Researchers looked at the usefulness of many representations in the Bidirectional Long Short-Term Memory (bi-LSTM) model, compared them to 22 languages in varied circumstances, and suggested a new bi-LSTM model with auxiliary loss for POS tagging in (Plank, Søgaard, & Goldberg, 2016) Their context-based bi-LSTM, which employs word embeddings as inputs, is the most basic variant of the model. They also employed sub token level embedding of words at a lower level (character or Unicode byte), which was concatenated with word embedding before being fed into the context bi-LSTM. They train their proposed model tagger to predict not only the sequence's tags, but also a label that shows the log frequency of each tag's occurrences.

One of the numerous instances of articles that expand on the high accuracy findings of LSTM application on POS tagging is the study (Lőrincz, Nuţu, & Stan, 2019), which is only one of the many examples. The authors of this study report that accuracy ratings of around 99% and 98% were achieved for two distinct forms of POS expressions in the Romanian language, respectively. In addition, the authors state that the method is language independent, which indicates that the application of LSTM for POS tagging in any language would show the same amount of accuracy scores.

The subsequent paper (Bahcevan, Kutlu, & Yildiz, 2018) presented an examination of PoS tagging in the Turkish language. In the area of neural network language modeling and word embedding for the Turkish language, the authors are pioneers. They conducted research and made contributions to these two fields. The authors of this paper develop the Long Short-Term Memory (LSTM) technique to conduct POS labeling for the Turkish. As a direct result of this, they have discovered that LSTM received an F-1 grade of 88.7 % on the task that was given to

them. Nonetheless the LTSM surpasses the RNN in terms of the f1-score metric, the LSTM method is insufficient to compete with . It is preferable to explore diverse algorithms and evaluate them to against LSTM in order to get best results.

This research (Toleu, Tolegen, & Mussabayev, 2020) outlines four different neural network algorithms for part-of-speech tagging by applying them to a range of languages, including Turkic languages, which have unique typological characteristics. Experiments conducted for this study show that multilingual PoS tagging utilizing the LSTM with CRF layer outperforms the other three approaches.

To create a POS tagger for Hindi, (Singh, Verma, Seal, & Singh, 2019) presented deep learning methodologies. They used a huge dataset of 50,000 tagged texts to test their theory. According to the results of the experiment, the suggested model attained an average tagging accuracy of 97.05 %. The research utilizes a manually tagged dataset for training and makes no comparisons to earlier studies.

A POS tagger for Nepali that is based LSTM, Bi-LSTM, RNN, and GRU was developed in (Sayami, Shahi, & Shakya, 2019). The models are trained and evaluated on the corpus of Nepali language; thus, with a testing accuracy of 97.27 % the performance of the Bi-LSTM algorithm is superior to that of the other three approaches. One of the problems with this research is that in contrast to earlier studies, the researchers employ smaller corpus for training and evaluation purpose. Moreover, their works not compared to the results of previous ones.

(Deshmukh & Kiwelekar, 2020) developed POS labeling in favor of Marathi language text using a bidirectional long short-term memory (Bi-LSTM). They attempted to build POS annotator models using Bi-LSTM on three-fold validation. Bi-LSTM obtained an accuracy of 97 % in the experiment. This dataset is also used to evaluate machine learning approaches including Bayesian inference, Hidden Markov models, KNNs, random forests and conditional random fields to the suggested Bi-LSTM and other deep learning networks. Experiments are run using 1500 sentences totaling 10,115 words, which is a much lesser amount than what is often used in order to form Bi-LSTM and deep learning approaches. In addition, the suggested approaches are not evaluated against the most recent and cutting-edge research in the same subject of research.

The POS tagging in the Malayalam language was accomplished by (Akhil, Rajimol, & Anoop, 2020) using techniques based on deep learning. They used the publicly accessible Malayalam

corpus that had an around 290 000 labeled words in addition to the tag set of 36 notation from the Bureau of Indian Standards (BIS). They tried out a variety of sequential deep neural networks, including long short-term memory (LSTM), gated recurrent unit (GRU), and Bi-LSTM, among others. The trials were carried out using hidden layers of 4, 16, 32, and 64 respectively. An f-measure of 98 % was said to have been attained by the Bi-LSTM model with 64 hidden layers. They are using f1-score, recall and precision as measurement of performance and efficiency of their proposed model, however it would be better to evaluate with accuracy score also.

The authors (Gopalakrishnan, Soman, & Premjith, 2019) developed and tested a POS tagger for the biomedical sector which is based on a deep neural network. Throughout the course of carrying out the experiment, the LSTM, RNN, and GRU algorithms were utilized by them. The tagger is analyzed using three different approaches the LSTM, RNN, and GRU also their bidirectional variants, in order to develop a model which has to be more effective than previous studies. A study demonstrates that straightforward unidirectional sequence models, LSTM, RNN, and GRU achieve less level of accuracy than more complex bi-directional versions. The performance of these algorithms was improved because they are able to acquire and comprehend a greater quantity of information about terms from dataset. The experimental results for the suggested model have reached an accuracy rate of 94.80% for detecting proper parts of speech tags. The proposed model is not evaluated in light of the relevant past studies that have been done to advance the state-of-the-art in the field. Additionally, not conducting tests using other algorithms, some of which could provide results that are superior to the presented model, may be considered one of the cons to the study.

(Patoary, Kibria, & Kaium, 2020) recommended development of deep learning-based POS labeling methodology in favor of the Bengali language in their study where the suffixes of the language are serving as the primary source. To perform the study, researchers established an annotated dataset comprising 2927 words. A precision rate of 93.90% was reached by the deep learning-based POS tagger, which also outperformed earlier models including global linear and rule-based ones. Furthermore, the suggested model has been implemented in Python as a part of the open-source Bengali NLP library. One of the things that might be improved about this study is that the corpus that was utilized for the experiment probably isn't adequate to properly simulate deep learning. The accuracy score is used as the only criterion for determining how

well the suggested solution performs. Therefore, the efficiency of the model could change when it's measured using other quality determiners like f1-score, recall, and precision.

For the purpose of providing superior outcomes with regarding to speed and accuracy, a Feedforward Neural Network (FNN) technique has been suggested for character-level word encoding (Anastasyev, Gusev, & Indenbom, 2018). Moreover, they set up loss functions as a pattern to comprehend dependencies. They used three datasets: Vkontakte, news, modern literature and achieved 95.64%, 96.46%, and 97.97% accuracy scores. One of the shortcomings of this study is employing only accuracy score where participants could make use of other performance measurements. Also, their model did not perform significantly well than relevant studies.

(Besharati, Veisi, Darzi, Hosseini, & Seyed, 2021) introduce hybrid approach for Persian language, which is combination of multi-layer perceptron (MLP) and long short-term memory (LSTM) for assigning appropriate parts of speech annotation to in and out of vocabulary terms. Proposed hybrid technique is effective in enhancing both Hidden Markov Model and neural networks, bringing the level of precision up to 97.29 %. In the study they have advanced alerts and minor bugs related to extracting word vectors.

Chapter 3

Methodology

3.1 Hidden Markov Model (HMM)

The use of probability in the tagging of tasks is a realistic option since it estimates how probable it is for a word to be a certain element of speech based on the insights obtained from a large corpus of text. Because the syntactic properties of a language are taken into consideration while carrying out a job, this method generates exceptionally good results. For instance, in certain varieties of the language, the verb comes before the noun, whilst in others, the pronoun is used rather than the noun. The Hidden Markov Model is a kind of model that takes into consideration the likelihood of states that, due to the presence of tags, are tags, as well as the chance of advancing between these states. Figure 2 provides a clear and concise illustration of the operational criteria as well as the formalization of HMM in speech labeling using models for the English language. During the course of the project, this illustration has been accepted as a rule for understanding and properly using HMM for speech tagging.

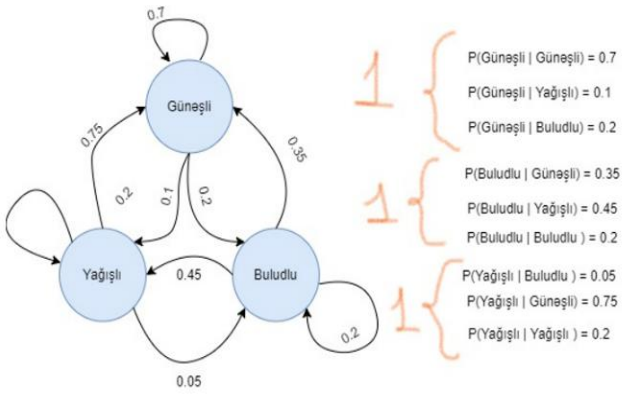

Figure 2. Graphical representation of HMM

HMM for PoS tagging is dependent on the Bayesian inference, which is a method in measurable deduction that uses Bayes' hypothesis to determine the probability of an event given another event. Bayesian inference is thus a technique in measurable deduction. PoS tagging may be seen more closely as a classification problem due to the fact that the words in a specific book

are organized into a variety of tag classes. Because the order in which tags are applied is also taken into consideration while doing the tagging, this issue might be categorized as a sequence classification challenge.

Let's look at an example and then go on to the next step so that we can better understand the HMM rules. For example, if we have a sentence that reads “Dissertasiya işini yazmaq həqiqətən də bu qədər çətindirmi?” which is translated into English as “Is it really that difficult to write dissertation?” how does HMM determine the most probable sequence of tags to go with it? According to Bayesian inference, each and every conceivable sequence of tags has to be taken into consideration, and from among these sequences, the one that has the greatest probability is selected as the sequence that should be followed. The following formula is used to determine which tag sequence has the greatest probability:

$$\hat{t}_1^n = argmax\,(t_1^n|w_1^n)$$

Given a string of words, this function selects the order of tags that has the highest likelihood of being correct. In order to compute this probability, we make use of Bayes' rule. The concept that underpins Bayes's theorem is to transform a probability that relies on the knowledge of several esoteric probabilities into a link of various probabilities that are already common knowledge. The equation that represents Bayes's theorem may be formulized as follows:

$$P(X|Y) = \frac{P(Y|X)P(X)}{P(Y)}$$

Changed to:

$$\hat{t}_1^n = argmax\frac{P(w_1^n|t_1^n)P(t_1^n)}{P(w_1^n)}$$

When considered in light of the present issue that we are facing. The likelihood of coming across a selected word in a corpus is represented by the value in the formula's denominator. Since the probability does not vary regardless of the total number of particular words or the size of the corpus, we may ignore the denominator in the calculation and arrive at the following result:

$$\hat{t}_1^n = argmaxP(w_1^n|t_1^n)P(t_1^n)$$

After simplifying the equations, the HMM tagger takes into account the fact that the likelihood of a particular word appearing in a corpus does not rely on other words but rather relies only on the word itself.

$$P(w_1^n|t_1^n) = \prod_{i=1}^{n} P(w_i|t_i)$$

Taggers are able to be classified based on their purpose, which is to assign the word based on their placements. Since the unigram tagger does not take into account the likelihood, there is no sequence in this particular instance. In this project, the bigram and trigram tagging techniques are used. These approaches take into account the tag that came before them when they tag the next word. In bigram tagger, we compute probabilities by multiplying the probabilities of tags given the tags that came before them. This allows us to get a more accurate picture of the situation.

$$P(t_1^n) = \prod_{i=1}^{n} P(t_i|t_{i-1})$$

Because of the high number of homonyms found in the Azerbaijani language, the tagger may be able to identify the tag of a word by looking at the tag of the word that came before it. This bigram tagging method makes the task more accurate. Because of this fact, we may deduce that the tagger has the ability to pick whether the word "alma" – "don't buy" refers to a verb or a noun by determining if the word that comes before "alma" – "apple" is an adjective or a noun.

$$P(t_i|t_{i-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$

The process of decoding is also considered to be one of the most significant aspects of HMM models. Finding the hidden variable sequence is part of the decoding process, and so is identifying the tag sequence. When working with HMM, the Viterbi algorithm is the most helpful decoding technique to apply Figure 3.

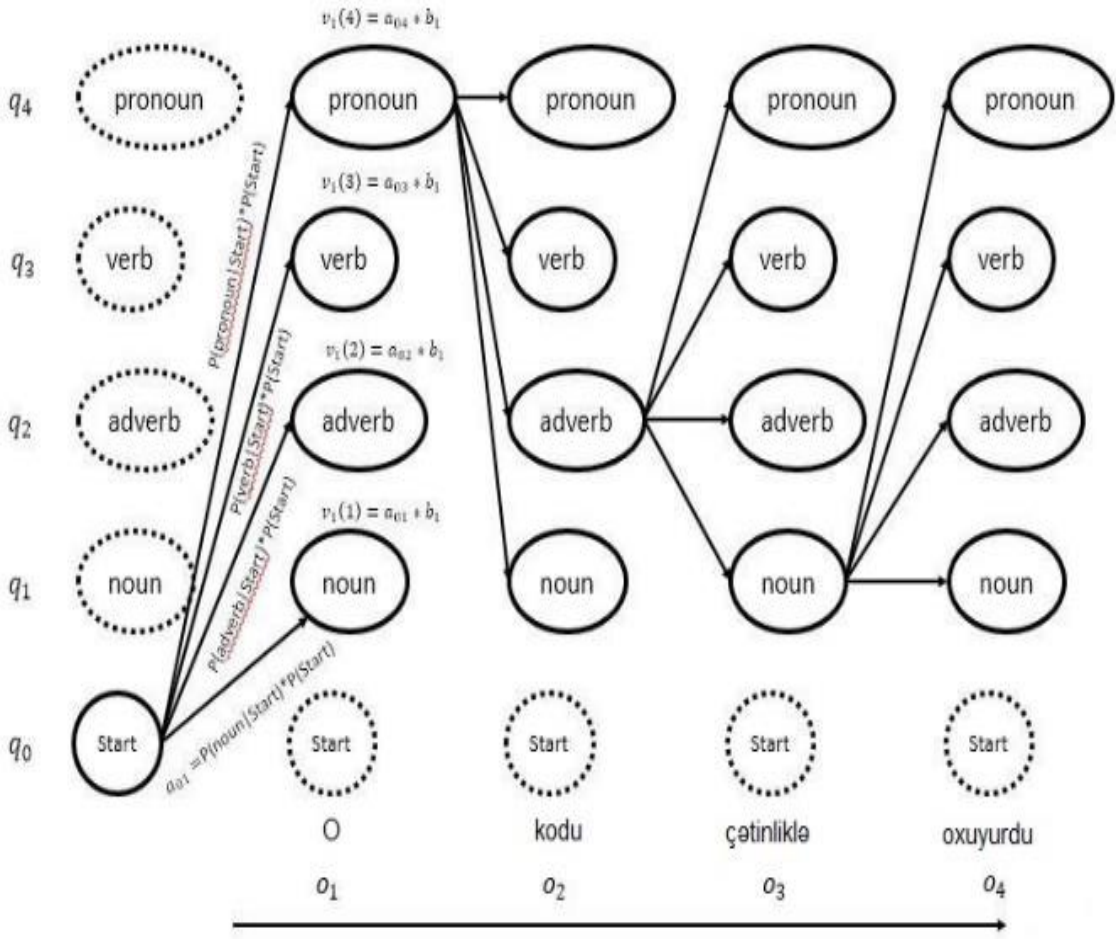

Figure 3. Representation of Viterbi Algorithm on Azerbaijani language

3.2 Conditional Random Field (CRF)

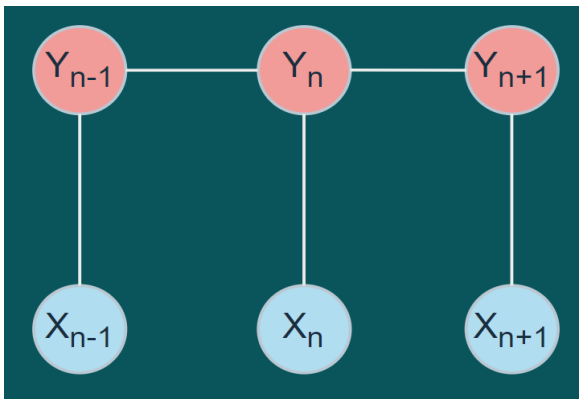

Figure 4. Graphical representation of linear CRF

In the context of predictive tasks, Conditional Random Fields (CRF) is a type of stochastic approaches that are best suited. We employ characteristics that are derived from the data to insert into the CRF since these paradigms take into consideration preceding inputs. The tag sequence "Sifət" → "Isim" → "Fel" is an example of a feature function that illustrates a property of the sequence that the data point represents. In terms of mathematic formula as described below, the likelihood of a given tag sequence or hidden state y and specified observation variable x is being a normalized $\mathbb{Z}(x)$ (Lafferty, McCallum, & Pereira, 2001):

$$P(y|x) = \frac{1}{\mathbb{Z}(x)} \prod_{t=1}^{T} \exp\{\sum_{k=1}^{k} \theta_k f_k(y_t, y_{t-1}, x_t)\}$$

To convey certain aspects of the training data, we generate a collection of real valued features $v(x,j)$ of the observation. If the present state or the current and previous states take on certain specific values, each feature function takes on the value of one of the real-valued observation features $v(x,j)$. As a result, in nature, all feature functions have a real value as described at below:

$$f_t(y_j, y_{j-1}, x, j) = \begin{cases} v(x,j) & if\ y_{j-1} = Isim\ and\ y_j = Fel \\ 0 & otherwise \end{cases}$$

$f_t(y_j, y_{j-1}, x, j)$ can be state or transition function. During the training phase, they are calculated by maximizing the conditional log probability on a collection of samples that have previously been labeled (training data) (Constant & Sigogne, 2011). The process of decoding involves labeling a new input sequence with regard to the model in order to achieve maximum $P(x|y)$. This is done in order to complete the operation (or minimizing $-\ log\ P(x|y)$). In order to rapidly investigate all of the possible labeling configurations, dynamic programming approaches such as the Viterbi algorithm are at your serve.

3.3 Recurrent Neural Network (RNN)

It is possible to expand what is known as a feed-forward neural network (FNN) into something that is termed a recurrent neural network (RNN), which has an internal memory. RNN is said to have a recurrent characteristic due to the fact that it applies the same function to each and every piece of data that it accepts as input, and the output of the most recent set of data is dependent on the computation that was carried out on the most recent set of data. As soon as

the output is produced, a duplicate of it is constructed and then fed back into the recurrent network. This happens immediately. When it comes time to make a decision, it takes into consideration both the input that it is receiving at the moment as well as the output that it has gotten as a result of the information that it has obtained in the past.

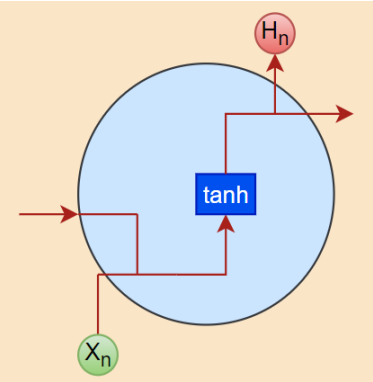

Figure 5. Vanilla RNN architecture

RNNs, as opposed to FNN, have the ability to process sequences of inputs by using their internal state, often known as memory. Because of this, they are suitable for applications such as voice recognition and unsegmented, linked handwriting recognition. In different neural networks, each of the inputs is considered to be autonomous from the others. However, with RNN, all of the inputs are connected to one another in some way.

The calculation of fixed-size vector representations for word sequences of arbitrary length is made possible by recurrent neural networks, also known as RNNs (Elman, 1990). A recurrent neural network (RNN) is a function that takes in n vectors $(x_1, x_2 \dots x_n)$ as input and creates an output vector (h_n) that is dependent on the full sequence $(x_1, x_2 \dots x_n)$ of input vectors (Plank, Søgaard, & Goldberg, 2016). After that, the vector (h_n) is used as an input to some classifier, or higher-level RNNs, in models that are stacked or hierarchical. The whole network is trained together in such a way that its hidden representation is able to accurately capture the essential elements of the sequence for the purpose of the prediction job.

$$h_n = f(x_n, h_{n-1})$$

The above formula shows that the current tag t_1 is influenced by the preceding k tags. The past output is employed to forecast the upcoming one in this case.

Here h_n serves two purposes:

- first, it will generate an output prediction
- second, it will maintain a hidden state that represents the data sequence that has been processed up to this point.

The present input is indicated by the notation x_t, and the timestep that came before it is indicated by the notation h_{n-1}. Computation of the prediction for the hidden state requires both of these notations. RNN are afflicted with two distinct problems as a result of the fact that it "fails to recall" initial x variables: (1) Vanishing gradient and (2) Exploding gradient.

In deep neural networks, there is a theory that states the accuracy of the model could improve with an increase in the number of hidden layers (Shewalkar, 2018). This sort of network is capable of gleaning additional information from the data it processes. In order to train a neural network that is this deep, the approach that we are using is called stochastic gradient descent by backpropagation. It has come to our attention that the various layers of our deep neural network are picking up information at varying rates. When adding more hidden layers, there is a chance that the accuracy may decrease. This means that the classification accuracy will continue to decrease as more layers are added to the network. The issue is that our learning system is unable to locate the appropriate weights and biases for the situation. If we continue to add new hidden layers, the earlier hidden levels will learn at a much slower rate than the later hidden layers (Gopalakrishnan, Soman, & Premjith, 2019).

3.4 Long Short-Term Memory (LSTM)

(Hochreiter & Schmidhuber, 1997) advocated the use of an LSTM cell as a solution to the issue of "long-term dependence." By incorporating a "gate" into the typical recurrent cell, the researchers were able to increase the ability of the cell to recall information. Since the publication of this ground-breaking study, LSTMs have been refined and made more widely used by a number of academics. There are many variants of the LSTM, the most common of which are with a forget gate, the LSTM without a forget gate, and with a peephole connection. In most contexts, the phrase "LSTM cell" refers to an LSTM that also includes a forget gate (Yu, Si, Hu, & Zhang, 2019). As shown in Figure 6, the formation of an LSTM network is quite similar to that of a regular RNN; however, in replacement of the self-connected hidden layers that are typical of RNNs, an entirely new concept known as a memory block is used. According to this article (Graves, Mohamed, & Hinton, Speech recognition with deep recurrent neural

networks, 2013), the following composite function is used to calculate the output of the LSTM hidden layer h_n given the input x_n:

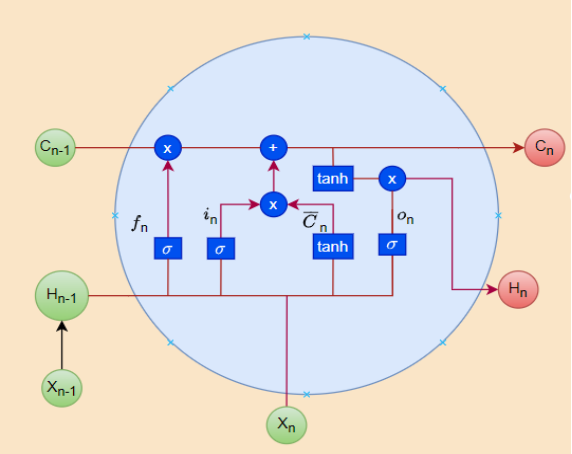

Figure 6. LSTM cell

$$i_n = \sigma\,(W_{xi}x_n + W_{hi}h_{n-1} + W_{ci}c_{n-1} + b_i)$$

$$f_n = \sigma\,(W_{xf}x_n + W_{hf}h_{n-1} + W_{cf}c_{n-1} + b_f)$$

$$c_n = f_n c_{n-1} + i_n tanh(W_{xc}x_n + W_{hc}h_{n-1} + b_c)$$

$$o_n = \sigma\,(W_{xo}x_n + W_{ho}h_{n-1} + W_{co}c_n + b_o)$$

$$h_n = o_n\, tanh(c_n)$$

- σ corresponds sigmoid activation function from logistic regression in machine learning which is responsible for the forcing inputs to be range of 0 and 1
- i sign for input gate: this is still another sigmoid layer, determines which parameters need to be adjusted.
- f illuminates forget gate notation which informs us what we need to block out of our memories.
- c illustrates network cell, updated when
- o shows output gate symbol, uses a sigmoid function to determine what should be output.
- h provides a vector of new candidate values.

3.5 Gated Recurrent Unit (GRU)

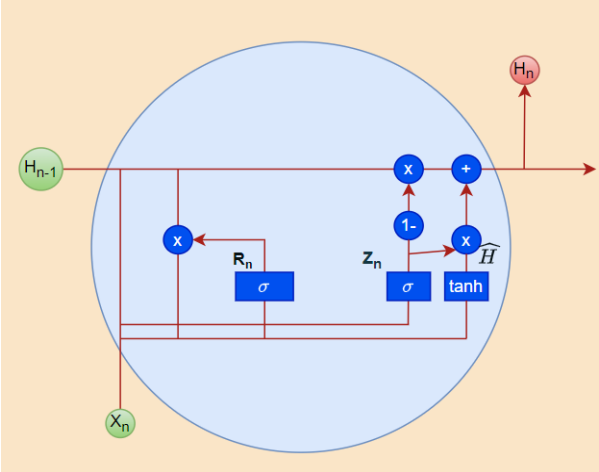

Figure 7. GRU CELL

The typical recurrent cell does not equal to the superior standard of learning ability of the long short-term memory (LSTM) cell. However, the increased computing overhead is a consequence of the added parameters. As a result, (Cho, et al., 2014) came up with the concept of the gated recurrent unit, also known as the GRU. The GRU cell's intricate design and network of connections are shown out in full in Figure 7.

$$r_n = \sigma\left(W_r x_n + U_{hr} h_{n-1} + b_r\right)$$

$$z_n = \sigma\left(W_z x_n + U_z h_{n-1} + b_z\right)$$

$$\hat{h} = \phi_h\left(U_{\hat{h}x} x_n + U_{\hat{h}h}(r_n \cdot h_{n-1}) + b_h\right)$$

$$h_n = z_n \cdot \hat{h}_n + (1 - z_n)\, h_{n-1}$$

As a you can see from formulas above, it means of cutting down on the total number of parameters, the GRU cell utilizes the LSTM cell's forget gate and input gate, combining them into a single update gate (Yu, Si, Hu, & Zhang, 2019). Only two gates: an update gate denoted by the notation z_n and a reset gate denoted by the notation r_n are present in a GRU cell. As a consequence of this, it is feasible that it will be possible to store a single gating signal together with the parameters that are associated with it. The GRU is just an extended form of the standard LSTM that includes a forget gate. The power of the single GRU cell is lower than that of the first LSTM since it only has one gate instead of two.

3.6 Bidirectional Long Short-Term Memory (Bi-LSTM)

Utilizing information from the sequence's previous (on the left) and subsequent (on the right) steps becomes beneficial in many cases when the steps are organized sequentially. The normal LSTM architecture, on the other hand, is only aware of the results of computations that have come before it and is completely clueless about what will come next in the sequence. Therefore, the use of a bidirectional LSTM, also known as BLSTM, offers an elegant solution to this issue. The plan is to calculate the sequence from front to back and back to front in order to get knowledge about the past and the future (Ma & Hovy, 2016). The BLSTM networks are often more powerful than the LSTM networks (Graves & Schmidhuber, 2005).

BLSTM, developed by (Schuster & Paliwal, 1997) increases the network's capacity for receiving input data. Neurons may be classified as either forward- or backward-firing. Thinking about this sentence, "Mən – Pronoun məktəbə - Noun gedirdim - Verb ancaq – Conjuctive yolda – Noun yaranan – Practile problemə-Noun görə-Postposition geri-Adverb qayıtdım-Verb" – "I was going to school, but I returned due to a problem on the way," we want to predict appropriate parts of speech tag the term "ancaq." A unidirectional LTSM will only store information about "Mən məktəbə gedirdim ancaq" is "Pronoun, Noun, Verb ..." to predict while Bi-LTSM will have information about tags of "yolda yaranan problemə görə geri qayıtdım" which is "... Noun, Practile, Noun, Adverb, Verb," so which one is going to predict better? Of course we can supervise that Bi-LTSM will understand context better than unidirectional LTSM as it has more information about the data, and will return "Conjunctive – Bağlayıcı" as answer.

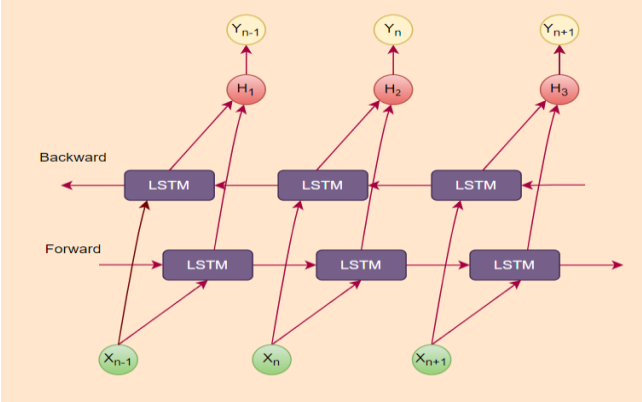

Figure 8. Bi-LSTM architecture

Chapter 4

System's Architecture

Summary

Throughout this chapter, we will discuss our proposed design and foundational considerations parts of speech for tagging Azerbaijani language. Deep learning, which is now the most discussed topic in the field of machine learning research and development, is used in this design. On the other hand, deep learning algorithms are able to build features from input in a more efficient manner than the other built features that are used by these approaches. The recommended approach for developing a parts-of-speech tagger for the Azerbaijani language makes use of the following network architectures: Recurrent Neural Networks (RNN), Gated Recurrent Units (GRU), Long Short-Term Memories (LSTM), and Bi-directional Long Short-Term Memories (Bi-LSTM).

4.1 Data collection & Corpora

Because the vast majority of documents in existence today are written in a format that is readable by machines and are accessible over the internet, the construction of raw text corpora does not pose a significant challenge anymore. Of course, when assessed against wide-spread languages (e.g., English, Chinese) assembling raw data in Azerbaijani presents an issue that is somewhat more challenging. This probably would be common case for the other agglutinative low-resources languages. The real reason behind scene is a lot of encoding schemas are existing, which leads to this situation in turn. Even though most of our alphabetics is suitable for "utf-8", still can lead problems while working libraries like Pandas, Word2Vec. For example, letters "İ," "Ə," "Ğ," "Ü," "Ö" cause the problem when I was using to_csv() methods of Pandas library, and to solve that I changed encoding format. In addition, the amount of clean Azerbaijani documents that can be found on the internet is rather low when compared to other languages.

Hence, we decided to create a dataset will supply our experiment. We collected various short blogs, stories, and news in Azerbaijani language, we clean data to annotate by using proper tag set. We adopted design of Brown corpus for our dataset.

The Azerbaijani language does not have a publicly accessible annotated corpus, as far as we know. Even we do not need to mention the fact that there is no balanced corpus. As a result, the process of building a corpus with mixing categories utilizes an incremental method. This information is presented in the introduction section of the thesis. After raw texts are cleaned to be manually tagged by native speakers. The book "Explanatory dictionary of the Azerbaijani language" which is comprised of 4 volumes was used by native speakers while tagging. Our dataset contains around 20000 words from Azerbaijani language. This method is carried out again and again until the target size of the corpus for this thesis work, which is 1809 sentences, has been reached. Figure 9 demonstrates distribution of sizes of sentences in our data. The longest sentence in our dataset has 88 words. Large portion of corpus is consisting of sentences which have 1-40 words.

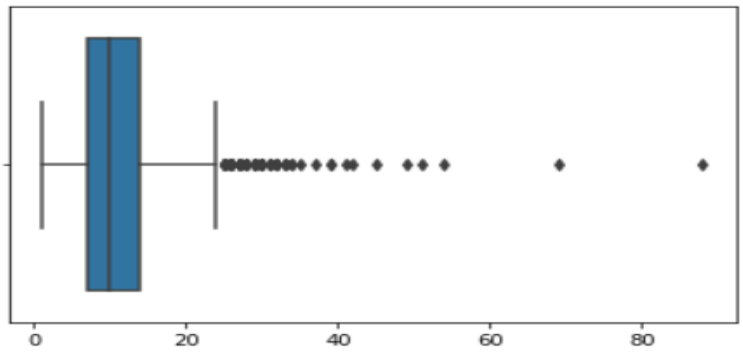

Figure 9. Distribution of sentences for Azerbaijani corpora

Because it requires data to be gathered from a variety of fields, the creation of a balanced corpus takes a significant amount of time, as well as the labor and expertise of language specialists. As a result, rather than developing a balanced corpus which has direct effect on results especially for stochastic approaches such as HMM, and CRF, in this thesis, a restricted category was chosen. News and report texts are readily accessible and can be gathered easily from a variety of sources rather than developing a balanced corpus, is also one of the reasons, we preferred this way. Moreover, in truth, the major objective of constructing a balanced corpus with the aim of rising the efficiency of the annotator when it labels any data chosen from random group in particular. This instantly suggests that a balanced corpus comprises as many words as feasible from each category using them in the sense that they were meant to be used. Also, the vocabulary and expressions that are most often used in a certain field would be included in a

corpus that is limited to only that particular field. If a text that has to be tagged from another category is presented to the tagger as training material, the performance of the tagger may deteriorate as a consequence. On the other hand, if the text that was selected came from that category, then it is presumable that the performance was the same as projected.

Both linguistic research and the subsequent automated NLP operations require to have their corpora annotated with syntactic information in the form of POS tags (Heid, Wever, & Hüllermeier, 2020). This is an essential necessity. It is usual practice to tackle this problem by employing the principles of machine learning, especially by training a POS annotator to operate on a suitably huge body of annotated data. Although the issue of POS labeling may typically be seen as addressed for contemporary languages, it turns out to be substantially harder for historical corpora, especially because there are fewer native speakers and there is less training data available. In addition, the great majority of works do not utilize a standard spelling or express sentences in the style we are accustomed to viewing them today. The process of automated POS tagging is rendered more complex and prone to error as a consequence of these variances. If the POS tagger can express its ambiguity, rather than forcing it to make a forecast and choose a single tag, it is desirable. Heid and Hüellermeier at (2020) investigate POS labeling across the context of list-valued guess. This gives the POS annotator the ability to convey its ambiguity by forecasting a list of potential POS labels rather than just predicting one. The objective is to minimize the number of potential candidates while simultaneously ensuring a high level of assurance that the proper POS tag has been included. Participants in this research found that enhanced cutting-edge POS annotators to list-valued guess resulted in extra precise and reliable labeling, particularly for unidentified terms, which are terms that did not appear in the training data. This was notably true for unfamiliar words. This was particularly true for unfamiliar words, or terms that did not be seen in the training data.

The most important aspect of the tagset from our point of view is its graininess, which is straight proportional to the overall scope of the corpus. In case the corpus is extra coarse, the labeling precision will be significantly greater because only the significant differences will be examined. Additionally, categorization may be simpler for both human manual annotators and the computer if the tagset is too coarse. However, because of the coarse granularity of the tagset, it is possible that some significant information may be omitted. Alternatively, a corpus that is extremely fine-grained can cause improvements in the data that is provided; however it will hinder the autonomous system's ability to execute. It's possible that POS tagger may decline.

When utilizing a fine-grained tag set, you will need to construct a model that is much richer in order to capture the encoded information; as a result, it will be more challenging to train. Even if we employ a tag-set with a very fine granularity, we won't be able to capture all of the subtle distinctions in POS tagging if we simply look at syntactic or contextual information, and often pragmatic level information as well.

In the course of our study, we have taken into account all 11 types of parts of speech that are found in the Azerbaijani language, and words with primary and auxiliary function. In addition to the parts of speeches discussed above, the tags that are utilized in the system may also be used to signify a variety of punctuation marks. The need of reaching better levels of precision in one's work is the driving force behind this endeavor. Punctuation marks add to the description of the context and make the application of the algorithm simpler by conveying particular indicators. This is because PoS tagging works by diminishing the importance of words based on the context in which they are used. We distinguished punctuations that end or divide sentences in Azerbaijan language which are followings: ".", "!", "?", ":", ";" and mark them as "/Durğu_işarələri". We choose it in order to make easy to achieve sentence tokenization. Furthermore, we had 3 forms of verbs that act as noun, adjective, and adverb, but it is easy to figure out them with stemmer since roots of all is verb and it is counting as verb for grammatical rule of Azerbaijan language. For example, if we say "Səni bura gətirən adam" it means "The person who brings you here," in this case, "gətirən" is participle and answering question "which man," since its root is "gətir" corresbonding "bring" it is not difficult to distimbiguate them. Also it is easy task for stemmer so it can handle it.

In consideration of all of the aforementioned grammatical constructions and punctuation marks, we have produced a total of 15 tags. Table 2 displays the descriptions of the tags together with the tags themselves.

The morphological richness of the Azerbaijani language is yet another essential characteristic of these languages. When determining the accuracy and performance of POS taggers, it is possible to take into consideration the level of morphological richness included in the data. A significant amount of agglutination may be found in the Azerbaijani language as it belongs all languages of Turkish family. The vocabulary grows in tandem with the dimensions of the dataset as it is expanded (the amount of unique words). (Mammadov, et al., 2018) proposed a stemmer for hmm type POS tagger by considering the deterministic characteristics of Azerbaijani languages. Researchers shared their solution for stemmer publicly. They gathered

300+ suffixes belong Azerbaijani language. While dig into their solution program which is written in python language, it could be easily seen that they touch issues related to grammatical structure of attaching suffixes for Azerbaijani language. In Azerbaijani, there are some suffixes which change the last syllable mostly last letter. As an example, “almaq” – “to buy” is being “almağın” when we are attaching suffix “in,” “q” is replaced with letter “ğ,” or “gəlmək” – “to come” becomes “gəlməyin,” “in” changed last letter “k” to the “y.”

№	Tags used in program	Example
1	/Isim=Noun	Çay=Tea
2	/Sifət=Adjective	Təhsilli=Educated
3	/Say=Numeral	Doqquz=Nine
4	/Fel=Verb	Fikirləşdi=Thought
5	/Zərf=Adverb	Bütün=Whole
6	/Əvəzlik=Pronoun	Bu=This
7	/Ədat=Particle (grammatical)	Yəni=I mean
8	/Modal=Modal	Beləliklə=So
9	/Bağlayıcı=Conjunctive	Hərçənd=Though
10	/Nida=Interjection	Vay=Ouch
11	/Qoşma=Postposition	Sarı=Towards
12	/Hissəcik=Particle	Idi=was
13	/Durğu_işarəsi=Puntctuations ends sentences	“.!;?:”
14	“,” = Comma	“,”
15	“-”= Dash	“-”

Table 2. Pos tags for our program

4.2 Data pre-processing

Even though Azerbaijani uses Latin alphabet when individuals text, particularly in the Azerbaijani language, they have a greater propensity to make frequent spelling errors in the words they type cause there are several different letters in Azerbaijani which cannot be found in English. It is due to popularity of English language, especially being for first choice for keyboards and people uses application in daily life for fun or in rush. Table 3 illuminates Azerbaijani alphabet letters and their potential usage on English keyboards.

Characters can be used alternative exchange							
ş	ç	ə	ı	ö	ü	ğ	İ
sh, s, w	ch, c	e, a	i	o, oe	u, ue	g, q	I

Table 3. Letters in Azerbaijani and possible expression in English keyboards

Second issue, since Azerbaijani language were adopted Cyrillic alphabet for long period of time, there are a lot of documents, papers can be found in Cyrillic. But we won't touch this problem as part of our solution since utilization of Cyrillic is nearly dead. However, researchers especially who wants to investigate applications of NLP historical data can see it as future problem need to solve.

Third thing to consider when data-processing as we mentioned before, the suffixes groups of the Azerbaijani language is consisting of two categories: lexical suffixes and grammatical suffixes. (Fatullayev, 2008) By combining word stems in a certain order, lexical and grammatical suffixes produce a wide variety of word-forms from the same word stem (for instance, it is possible to produce the word-forms "məktəb" – "school", "məktəb-də" – "at school", "məktəb-də-ki" – "at school", "məktəb-də-ki-lər" – "persons at school", "məktəb-də-ki-lər-dən" – "from persons at school", "məktəb-də-ki-lər-dən-siniz" – "are you one from school", "məktəb-də-ki-lər-dən-siniz-mi" – "are you one from school". In our proposed approach stemmer at (Mammadov, et al., 2018) slightly changed and adopted for our solution as base data pre-processing. First raw input text comes and stemmer stem from the suffixes then it encoded and vectorized then goes our deep leaning-based POS tagger.

4.3 Tokenization

The process of tokenization involves chopping up the raw text into smaller pieces. Then via tokenization process this data is broken up into terms and sentences that are knowing as tokens. The context may be understood better or a model for NLP can be developed with the assistance of these tokens. By calculating the sequence in which the terms appear, tokenization provides assistance in deciphering the context of the source data.

Following the completion of the pre-processing step, the corpus is handed over to the sentence tokenization module. Within this module, each and every labeled sentence from the corpus is retrieved. The sentence list is produced with the assistance of the sorted dictionary. The sentences are broken up using the key that comes with the dictionary. As a result of running this sentence tokenization module, you will get a list of tagged sentence lists. The labeled sentences are then separated into a testing, training, and validation slices. For sentence tokenization in our corpus, we are using help of tag "/Durğu_işarəsi." In Azerbaijani language, we have punctuations ".!?;," to end sentences.

4.4 Word Embedding

Deep learning models have been used to learn word embeddings, which are gaining popularity and may be useful in a variety of NLP applications (Bahcevan, Kutlu, & Yildiz, 2018). Facebook's research team has created an open-source, free, lightweight library called fastText (Joulin, Grave, Bojanowski, & Mikolov, 2017) for the purpose of learning text representations and text classifiers. A low-dimensional vector is created by summing the vectors corresponding to the words that are produced by an n-gram of a character appearing in the text, and this vector is utilized to represent a text in fastText. Word vectors for 157 languages (including Azerbaijani) (Grave, Bojanowski, Gupta, Joulin, & Mikolov, 2018) have been pre-trained using fastText on data collected from Common Crawl and Wikipedia by using CBoW with position-weights, in dimension 300, using character n-grams of length 5, a window of size 5, and 10 negatives. Binary and text versions of the word vectors may be obtained from the official website of fastText. For the purpose of this investigation, trained word vectors that make use of an extension of the fastText model have been applied to the Azerbaijani language. Trained vectors file has 4.2 GB size for Azerbaijani language. One of the pros of fastText is that even though word from your data is not on vocabulary of fastText, it still creates vector representation for this word because words are represented by sum of their substring.

4.5 Softmax Activation

Weight matrices are often used to describe the parameters that must be learnt in the aforementioned designs (Kumar, Kumar, & Soman, 2019). It is determined via a cost function what each of the parameters should be. A gradient-descent approach is used to reduce the inaccuracy in the cost function (with regard to parameters). Weights are updated when the cost function's error gradient ($-\frac{\partial E}{\partial w_i}$) decreases. In other words, the quantity of inaccuracy is changed whenever the weight parameter is subjected to any kind of change. The minus indicates that the amount of inaccuracy will become less as time goes on. Learning occurs as a consequence of the propagation of gradients based on mistake. A better categorization may be achieved using learned weights.

$$CE = -\sum_{i}^{C} t_i \, log(f(s)_i)$$

Cross-entropy loss function is often used for tasks like sequence classification and is defined as the formula above. Probability distribution $f(s)_i$ is the i^{th} predicted class's probability distribution, while real probability distribution t_i is true distribution. Measurement of the difference between expected and actual labels is known as the loss function. Softmax() receives the neural network's output layer. For testing, it offers the likelihood of tags relating to the term w_i. This is the calculated value of the tag for the word w_i :

$$\hat{t}_i = \frac{arg\ max}{l \in 1,2, \dots n} P_i(l|w_1, w_2, w_3, \cdots w_m)$$

4.6 Architecture of proposed Parts of Speech tagger

In this module, the real job of POS labeling, which is the course of action of determining the right annotation to use for a certain term, is carried out. This POS tagging model receives its input in the form of the testing characteristics. The output of this module includes the projected POS tag that corresponds to each individual word. The evaluation module is provided with both the predicted tags as well as the target testing tag with the aim of assessing the efficiency of the solution.

The high-level architecture of the POS tagging model of our solution for Azerbaijani language is demonstrated in Figure 10. This model's preprocessing module, sentence tokenizing module,

word tag separation module, and POS tagging module are all identical to classification-based POS tagging models.

However, the deep neural network module takes the role of the feature extraction module as well as the machine learning algorithm module, in other words, we are not using handcrafted features instead word vectors are created based on pre-trained model for Azerbaijani language by fastText word embedding library.

Our solution was inspired by (Akhil, Rajimol, & Anoop, 2020) works which is about creating POS tagger based on deep learning algorithms for Malayalam, which is part of Indian family group, and low-resource, agglutinative, highly inflectional language, as Azerbaijani language. They adopted Bi-LTSM model for their architecture and got high results.

In our study, Keras framework by Google is employed for Azerbaijani POS tagger, and program is written in Python language. RNN, LSTM, GRU, Bi-LSTM deep learning models utilized for training. Keras library is working with numbers not with words and tags. Therefore, after words and tags separated, each of them is indexed by assigning unique integers. Since Keras only deals with fixed size arrays, the most common longest sentence in the dataset, according to Figure 9 it is 40 words long, is taken as fixed size. Accordingly, a value ("0" as index and "-PAD-" as the corresponding label) is added, or if it sentences length longer than 40 words is cut off, to ensure the same size in other sentences. Maximum sequence length is set to 40 in our deep learning models, so it means every word were replaced with vector size of 40. Finally, before the training phase, the tag sequences were converted to the sequences of One-Hot Encoded tags because we had only 15 tag it would be easy to represent them as 1 and 0, and words list was trained by fastText vectors. Using dropout regularization, we began by establishing an initial dropout rate of around 20 %. This implies that throughout the training process, each neuron will have 20 % of its neurons randomly picked and disregarded at each update cycle. Because of the ease with which non-linear activation functions may be implemented, Rectified Linear Units (ReLU) activation ends up being an extremely valuable tool for hidden layers (Patoary, Kibria, & Kaium, 2020). The softmax function enables us to do the conversion from the outputs of the unit to probabilities for use in multiclass classification. If this is the case, we will need to use it. We came to the conclusion that the categorical cross-entropy loss function would be the best option. In conclusion, we have decided to use the Adam optimizer method for classification tasks in order to optimize and update weights that have been properly tuned.

Overall workflow for our tagger goes like that:

- We take data from manually tagged corpora for Azerbaijani language
- Then preprocessing of data is starts in this phase, we prepare our data for next step
- Then we separate data to lists of sentences
- Then we shuffle data and divide to train, test, and validation
- Next step we separate sentences from tags and encode tags with one-hot encoding of Keras and vectorize sentences by using FastText embedding.
- After all our train data goes deep learning models such as RNN, GRU, LTSM, and Bi-LTSM
- Then we employ ReLU activation function to reduce loss for each epoch in hidden layers
- With the intention of avoiding from overfitting, we added dropout rate 0.2
- Then output evaluated based on f1, recall, precision, and accuracy scores.
- Then saved model predict tag in test data
- Finally, we evaluated accuracy

In the evaluation phase, Since Keras does not support f1, recall, and precision, we add manually f1_m(), recall_m(), and precision_m() functions to estimate them.

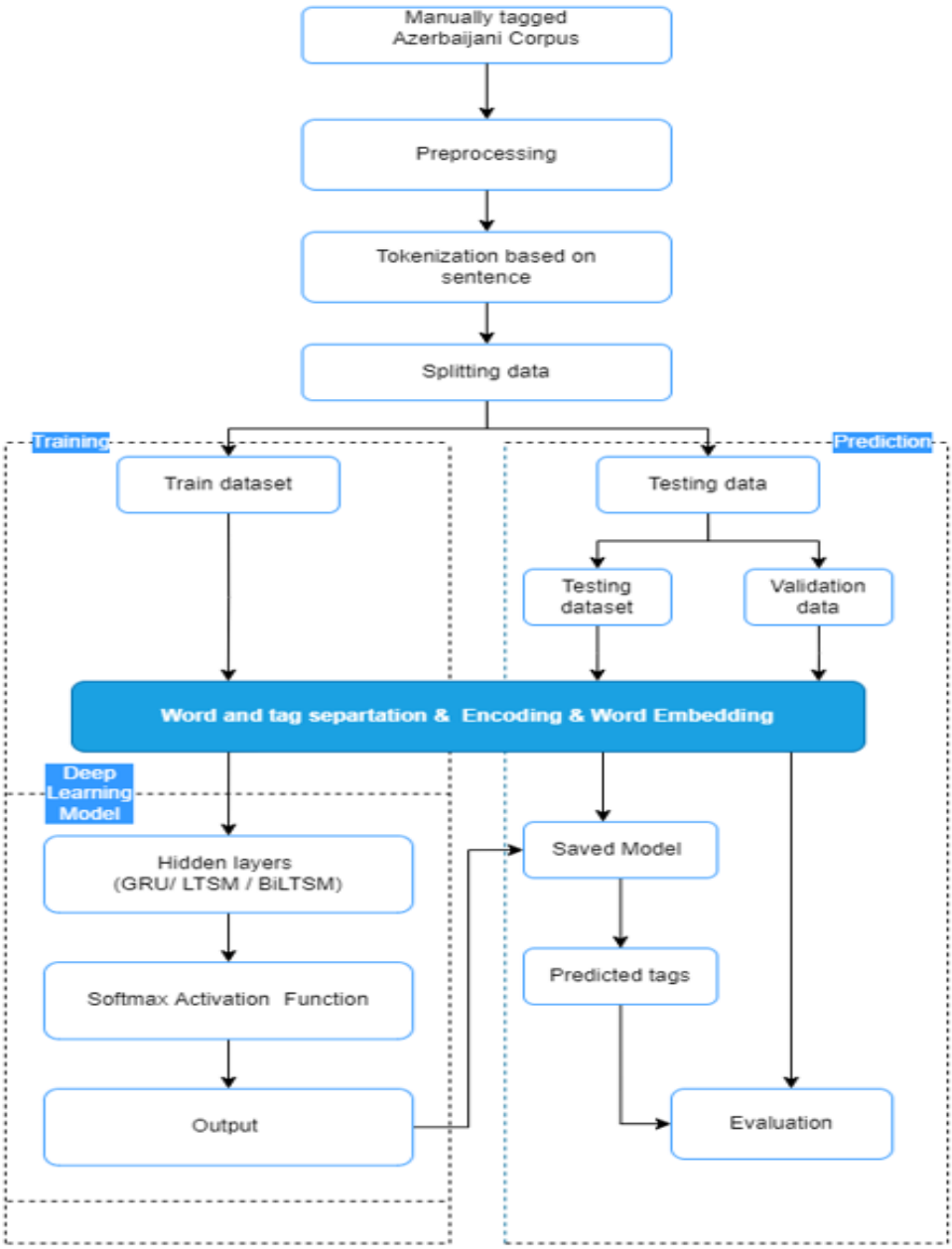

Figure 10. High-Level Design for proposed Part-of-Speech tagger for Azerbaijani language

Chapter 5

Experimental Results

Summary

Each experiment in this study is described in depth in this chapter, along with the findings that emerged from them. Each of the Parts-of-Speech tagging models described in the preceding chapters is put to the test performance. The studies were done using the manually tagged corpora created by us for Azerbaijani language POS tagger as described above. From the stochastic classification models, Hidden Markov Model (HMM) as suggested at (Mammadov, et al., 2018) and CRF are chosen and implemented as the best baseline model for Azerbaijani corpus. Moreover, the deep neural network models are contrasted with the baseline model to examine the efficiency of our proposed solution for Azerbaijani POS tagging over the classic machine learning methodologies. This study also conducts experiments with varying dataset sizes in order to examine how models respond to smaller datasets.

5.1 Experimental Setup

In this section, we will discuss the environmental setup for conducting experiments on our POS tagging solutions in Azerbaijani language by using deep learning models. In order to conduct tests, participants used free-of-charge for everyone product of Google Research's which is Colab notebooks. Google's Colab environment, which is open to the public and has a generic and completely adjustable Keras implementation and employs Google TPU processors for training and prediction (Civit-Masot, Luna-Perejon, Vicente-Diaz,, Corral, & Civit, 2019). For training our proposed model for Azerbaijani language, we decided to employ Keras top-level framework on TensorFlow machine learning toolkit and as background runtime environment we choose Google's hardware accelerator TPU with the purpose of getting full performance while training the suggested deep learning algorithms such as RNN, LSTM, GRU, and Bi-LSTM. Manually annotated corpora for the Azerbaijani language created by us and utilized for the test purposes. As it is first step for creating dataset for Azerbaijani language which tagged with appropriate parts of speech, we inspired by Brown corpus. Words of sentences in the corpus has followed proper tag expressed in Azerbaijani Table 2 and separated via slash "/". There are approximately 20000 words and 1809 sentences. The longest sentence has 88 words.

Majority of sentences is comprised of around 0-40 terms. Since we adopted coarse-grain style for our corpora for Azerbaijani language, tag set is consisting of 15 tags.

One layer was used for each and every one of our experiments, including the ones that made use of the RNN, LSTM, GRU, and Bi-LSTM algorithms. Following our deliberations, we have arrived at the verdict that the number of secret states in the future shall remain, respectively, 16, 32, and 64. There was a total of 5 iterations, also known as epochs, which were included in the first set of tests that we carried out. The activation function has been set to softmax and the size of the hidden layer has been set to 64 for each of the models that we have shown to you in the paragraphs that immediately precede this one. The network concluded that in order to accomplish the amount of progress that was required in the training, it should utilize a dropout parameter 0.2. This decision was reached after the network experimented with a number of different trial-and-error methods. Throughout the course of this particular investigation, we made use of a wide range of criteria for evaluating, some of which were precision, recall, and accuracy. We divide the dataset into three portions for the purposes of training, and 60% of the dataset is utilized for training. The remaining labeled words are split evenly into 20 **%** for each and are used for the purposes of testing and validation. A detailed breakdown of our findings, together with an analysis and discussion of those results, is presented in **Table 4.**

5.2 Evaluation Metrics

In this research we adopted 4 categories for analyze efficiency for our solution: (1) Accuracy (2) F1 score (3) Recall (4) Precision. In order to calculate them we need values of True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN). Let's look at following example to grab metrics that we selected evaluate, well. Since values we choose for the classification problems, it would be meaningless if we try to evaluate for words, that's way we will estimate it in terms of tags. Let's assume you have data consisting of 10 words and you have 6 nouns in this sentence. You have predicted 7 noun tags for this data. But out of the 7 noun predictions you made 3 mistakes, that means you mark 3 words as noun which are not noun. So, in this case your TP whichever you predicted true is 4 out of 7 where 3 out 7 predictions you thought this word would be noun and made mistake is called FP which is equal to 3. Now think otherwise you predicted number of words which are not noun is 3. You guessed 3 non-noun words and 1 of this prediction is true, and it is your TN. 2 out of 3 words you said it would not be noun, are noun, so your FN is 2. It is time to ask how many we guessed right? Answer for this question is 5 out of 10 which is your accuracy score.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{4 + 1}{4 + 1 + 2 + 3} = 0.5$$

Although, accuracy score says about how accurate our model predicts, it is not enough to assess our model's efficiency, cause what if we have 10 noun cases, and we predicted 10 nouns, so our accuracy will be 100%. That sounds weird since this is not case always. So, we need to consider another metrics when calculate. These are Precision and Recall metrics.

Your Precision is how many nouns you have got right out of all noun predictions:

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{4}{7} \approx 0.57$$

When we estimate Recall, we take truth samples as base, on other words, we have 6 nouns as truth samples. Your recall is how many you have got right out of all nouns?

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{4}{6} \approx 0.67$$

Till this time what we see was about noun prediction, then we need to ask ourselves what about not noun prediction, what is the precision and recall for it? So here F1 score comes to rescue us. It is a harmony for our Precision and Recall scores.

$$F = 2 \cdot \frac{P \cdot R}{P + R} = 2 \cdot \frac{0.57 \cdot 0.67}{0.57 + 0.67} \approx 0.61$$

Table 4 shows experimental results of 4 deep learning algorithms with different size hidden layers for Azerbaijani corpus.

Layers	Model	Recall	Precision	F1 score
16	RNN	0.67	0.68	0.67
	GRU	0.70	0.73	0.71
	LTSM	0.85	0.86	0.85
	Bi-LTSM	0.87	0.89	0.88
32	RNN	0.79	0.80	0.79
	GRU	0.89	0.90	0.89
	LTSM	0.91	0.93	0.92
	Bi-LTSM	0.93	0.99	0.95
64	RNN	0.87	0.88	0.88
	GRU	0.91	0.93	0.92
	LTSM	0.95	0.97	0.96
	Bi-LTSM	0.98	0.99	0.98

Table 4. Metrics for performance of Deep Learning Models for Azerbaijani language with 5 epochs

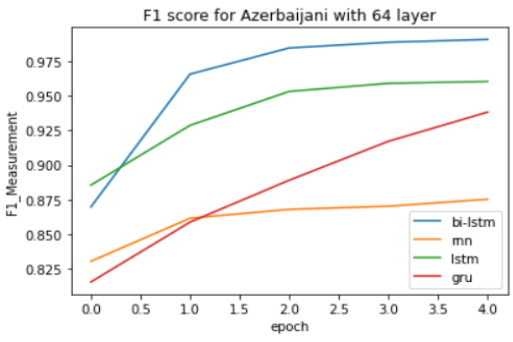

Figure 11. Comparison graph for f1_scores of deep learning algorithms on Azerbaijani language

Figure 11 clearly shows that Bi-LTSM outperforms all other models for Azerbaijani POS tagger.

5.3 Baseline Model

Hidden Markov Model POS tagger with stemmer for Azerbaijani language which uses Viterbi algorithm as proposed at (Mammadov, et al., 2018) and CRF chosen as a baseline system and developed by us with the purpose of assessing the effectiveness of our system, which is discussed in Chapter 3. They serve as a point of comparison for our findings when they are considered. The findings of the baseline systems may provide light on whether or not the primary model really contributes anything useful to the morphosyntactic categorization.

Approaches	F1_score	Accuracy
HMM	0.86	0.87
CRF	0.87	0.88
Bi-LTSM	0.97	0.98

Table 5. Accuracy Comparison of Models for Azerbaijani language

From the Table 5 we can see our solution to Azerbaijani POS tagger performs better than previous studies. Also, our dataset is 6 times bigger than corpus from previous study. We believe that parts of speech annotating model can be adopted by other Azerbaijani students, and computer science academics and for developing new NLP applications or improving current solutions.

Chapter 6

Conclusion

The primary goal of the study was to construct clean data corpora that contained both words and tag data and to determine which learning-based approaches perform the best on the datasets that were provided, with accuracy rates that were significantly higher than those achieved by other algorithms. Another objective of the study was to determine which learning-based approaches perform the best on the datasets that were provided. After the number of manually labelled phrases used for training reached 60 % clean datasets, the Bidirectional LSTM, CFR, and HMM algorithms were run on manually tagged data corpora, which had approximately 40 % unlabeled sentences, with the intention of evaluate the efficiency of the system. This was done with the target of determine how well the system worked. The findings of the trials revealed that, of all the algorithms that were used, the Bidirectional LSTM achieved the highest accuracy score (98 %) on both datasets. This was the case regardless of which way the data was input. The utilization and optimization of PoS tagging system using CRF and other deep learning algorithms in accordance with the linguistic approaches for the Azerbaijani language is what enables our paper to stand out among the other studies that have been conducted on this subject. It is of the utmost importance to point this out, as it is what enables our paper to distinguish itself from the other papers that have been written on this topic. This is something that should be brought up because of the significant impact it has. Furthermore, increasing the minimum accuracy rate for any implemented algorithm to 99 % through a better selection of parameters and integrating the best performing algorithm from text-to-speech technologies to the machine translation engines for the Azerbaijani language are both things that are being considered as potential future work. This is because both of these things are necessary in order to make progress. Both of these factors are relevant to the language that is spoken in Azerbaijan. In conclusion, it is very likely that the research that was carried out will make a contribution to the growth of Azerbaijani NLP systems and will create a basis for the research of part of speech labeling for other agglutinative languages. It could be something that is very exciting to think about. In addition, it is expected that these transformations will take place during the next several years.

References

Akhil, K. K., Rajimol, R., & Anoop, V. S. (2020). Parts-of-Speech tagging for Malayalam using deep learning techniques. *International Journal of Information Technology, 12(3)*, pp. 741-748.

Altunyurt, L., Orhan, Z., & Gungor, T. (2007). Towards combining rule-based and statistical part of speech tagging in agglutinative languages.

Anastasyev, D., Gusev, I., & Indenbom, E. (2018). Improving part-of-speech tagging via multi-task learning and character-level word . *Komp'juternaja Lingvistika i Intellektual'nye Tehnol. (17)*, (pp. 14–27).

Bahcevan, C. A., Kutlu, E., & Yildiz, T. (2018). Deep neural network architecture for part-of-speech tagging for turkish language. *3rd International Conference on Computer Science and Engineering (UBMK)*, (pp. 235-238).

Besharati, S., Veisi, H., Darzi, A., Hosseini, S., & Seyed, H. (2021). A hybrid statistical and deep learning based technique for Persian part of speech tagging. *Iran Journal of Computer Science, 4(1)*, 35–43.

Brill, E. (1995). A simple rule-based part of speech tagger.

Brill, E. (1995). Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. *Computer Linguistic*, 543–565.

C. D. Santos, & Zadrozny, B. (2014). Learning Character-level Representations for. *In Proceedings of the 31st International Conference on, 32*, pp. pp. 1818-1826.

Can, B., & Bölücü, N. (2019). Unsupervised joint PoS tagging and stemming for agglutinative languages. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP), 18(3)*, pp. 1-21.

Carneiro, T., Da Nóbrega, R. V., Nepomuceno, T., Bian, G. B., De Albuquerque, V. H., & Reboucas Filho, P. P. (2018). Performance analysis of google colaboratory as a tool for accelerating deep learning applications. *IEEE Access, 6*, 61677-61685.

Cass, S. (2019). Taking AI to the edge: Google's TPU now comes in a maker-friendly package. *IEEE Spectrum, 56(5)*, pp. p. 16-17.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation.

Civit-Masot, J., Luna-Perejon, F., V.-D. S., Corral, J. M., & Civit, A. (2019). TPU cloud-based generalized U-Net for eye fundus image segmentation. *IEEE Access, 7*, 142379-142387.

Collobert, R., Jason, W., Léon, B., M. K., Koray, K., & a. P. (2011). Natural language processing (almost) from scratch. *Journal of machine learning research, 12*, pp. 2493-2537.

Constant, M., & Sigogne, A. (2011). MWU-aware part-of-speech tagging with a CRF model and lexical resources. *In Proceedings of the workshop on multiword expressions: from parsing and generation to the real world*, 49-56.

Deshmukh, R. D., & Kiwelekar, A. (2020). Deep learning techniques for part of speech tagging by natural language processing. *2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, 76-81.

Elman, L. J. (1990). Finding structure in time. *Cognitive science, 14(2)*, 179–211.

Fatullayev, A. (2008). Overcoming Agglutination Difficulties in the Development of an MT system from the Azerbaijani Language. *Speech and Language Technology*.

Getachew, M. (2001). Automatic part of speech tagging for Amharic language an experiment.

Gopalakrishnan, A. P., Soman, K., & Premjith, B. (2019). Part-of-Speech Tagger for Biomedical Domain Using Deep Neural Network Architecture. *10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 1-5.

Grave, E., Bojanowski, P., Gupta, P., Joulin, A., & Mikolov, T. (2018). Learning Word Vectors for 157 Languages. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation.*

Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural networks, 18(5-6),*, 602-610.

Graves, A., Mohamed, A. R., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. *In Proceedings of ICASSP*, (pp. 6645–6649).

Hakkani-Tür, D. Z., Oflazer, K., & Tür, G. (2002). Statistical morphological disambiguation for agglutinative languages. *Computers and the Humanities, 36(4)*, pp. 381-410.

Heid, S., Wever, M., & Hüllermeier, E. (2020). Reliable part-of-speech tagging of historical corpora through set-valued prediction.

HIRPSSA, S., & Lehal, G. S. (2020). POS Tagging for Amharic Text: A Machine Learning Approach. *INFOCOMP: Journal of Computer Science, 19(1)*, p. 11-17.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9(8)*, p. 1735–1780.

Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2017). Bag of Tricks for Efficient Text Classification. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2*, (pp. 427-431).

Jung, S., Lee, C., & Hwang, H. (2018). End-to-end Korean part-of-speech tagging using copying mechanism . *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP), 17(3)*, p. 1-8.

Jurish, B. (2003). *A hybrid approach to part-of-speech tagging.*

Kumar, S., Kumar, M. A., & Soman, K. (2019). Deep Learning Based Part-of-Speech Tagging for Malayalam Twitter Data (Special Issue: Deep Learning Techniques for Natural Language Processing). *Journal of Intelligent Systems, vol. 28, no. 3*, 423-435.

Kurfalı, M., Üstün, A., & Can, B. (2016). Turkish PoS Tagging by Reducing Sparsity with Morpheme Tags in Small Datasets. *In International Conference on Intelligent Text Processing and Computational Linguistics*, p. 320-331.

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *In Proceedings of the Eighteenth International Conference on Machine Learning*, 282-289.

Lőrincz, B., Nuţu, M., & Stan, A. (2019). Romanian Part of Speech Tagging using LSTM Networks. *IEEE 15th International Conference on Intelligent Computer Communication and Processing (ICCP)*, (pp. 223-228).

Ma, X., & Hovy, E. (2016). *End-to-end sequence labeling via bi-directional lstm-cnns-crf.*

Mammadov, S., & Rustamov, S., & Mustafali, A., & Sadigov, Z., & Mollayev, R., & & Mammadov, Z. (2018). Part-of-Speech Tagging for Azerbaijani Language. *IEEE 12th International Conference on Application of Information and Communication Technologies (AICT)*, (pp. p. 1-6).

Patoary, A. H., Kibria, M. J., & Kaium, A. (2020). Implementation of Automated Bengali Parts of Speech Tagger: An Approach Using Deep Learning Algorithm. *IEEE Region 10 Symposium (TENSYMP)*, 308-311.

Perez-Ortiz, J. A., & Forcada, M. L. (2001). Part-of-speech tagging with recurrent neural networks. *International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222), (3)*, pp. 1588-1592.

Plank, B., Søgaard, A., & Goldberg, Y. (2016). Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss.

Sayami, S., Shahi, T. B., & Shakya, S. (2019). Nepali POS Tagging Using Deep Learning Approaches. *EasyChair (No. 2073)*.

Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. . *IEEE transactions on Signal Processing, 45(11)*, 2673-2681.

Shewalkar, A. N. (2018). Comparison of RNN, LSTM and GRU on Speech Recognition data.

Singh, A., Verma, C., Seal, S., & Singh, V. (2019). Development of part of speech tagger using deep learning. *International Journal of Engineering and Advanced Technology, 9(1)*, p. 3384-3391.

Smagulova, K., & James, A. P. (2019). A survey on LSTM memristive neural network architectures and applications. *The European Physical Journal Special Topics, 228(10)*, (pp. 2313-2324.).

Toleu, A., Tolegen, G., & Mussabayev, R. (2020). Deep learning for multilingual pos tagging. *International conference on computational collective intelligence*, (pp. 15-24).

Valizada, A. (2015). *Development of mathematical and software applications for PoS tagging texts in Azerbaijani language.*

Wang, P., Qian, Y., Soong, F. K., He, L., & Zhao, H. (2016). Part-of-speech tagging with bidirectional long short-term memory recurrent neural network.

Wu, S., Roberts, K., Datta, S., Du, J., J. Z., S. Y., . . . & Xu, H. (2020). Deep learning in clinical natural language processing: a methodical review. *Journal of the American Medical Informatics Association : JAMIA, 27(3)*, p. 457–470.

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. *Neural computation, 31(7)*, 1235-1270.