School of Information Technology and
Engineering at the ADA University

School of Engineering and Applied Science
at the George Washington University

Quantitative and Semantic Analysis of National Azerbaijan Corpus

A Thesis
Presented to the Graduate Program of Computer Science and Data Analytics
of the School of Information Technology and Engineering
ADA University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Computer Science and Data Analytics
ADA University

By
Abdulla Asgarzade

April, 2023

THESIS ACCEPTANCE

This Thesis by: Abdulla Asgarzade
Entitled: *Quantitative and Semantic Analysis of National Azerbaijan Corpus*

has been approved as meeting the requirement for the Degree of Master of Science in Computer Science and Data Analytics of the School of Information Technology and Engineering, ADA University.

Approved:

(Adviser)	(Date)
(Program Director)	(Date)
(Dean)	(Date)

ACADEMIC INTEGRITY STATEMENT

"I affirm that this is my own work, I attributed where I used the work of others, I did not facilitate academic dishonesty for myself or others, and I used only authorized resources for my Thesis, per the ADA University Academic Integrity requirements. If I failed to comply with this statement, I understand consequences will follow my actions. Consequences may range from failing the course to expulsion from the program/university and may include a transcript notation."

Abdulla Asgarzade		24.04.23
(Full Name)	(Signature)	(Date: DD.MM.YY)

ABSTRACT

This master thesis focuses on the quantitative (statistical) and semantic analysis of the Azerbaijani language corpus. The research begins with the creation of the corpus by collecting textual data on Azerbaijani language. The corpus was composed of various types of texts, including fiction, non-fiction, news articles, and social media posts. The texts were pre-processed by removing stop words, punctuation marks, and special characters. The pre-processed texts were then tokenized and lemmatized, which is the method of breaking down words to their most basic form.

The corpus was analyzed using various quantitative techniques, such as frequency analysis, ngrams, concordance analysis, and word semantic similarity with different metrics. Frequency analysis involves counting the occurrence of words in the corpus, which helps identify the most commonly used words and their frequency. Ngrams analysis involves counting the frequency of pairs or triplets of words, which helps identify common collocations and phrases. Finding a word or phrase's context throughout the corpus using concordance analysis allows for the identification of a term's usage and meaning in various settings. Word semantic similarity analysis involves measuring the semantic similarity between words using different metrics, which helps identify the relatedness of words in the corpus.

The POS tagging of the corpus is the other major component of the thesis. Implemented and tested were two distinct Part-of-Speech (POS) tagging models using Hidden Markov Models (HMM) and Long Short-Term Memory (LSTM). The practice of marking the parts of speech of words in a phrase is known as POS tagging. The training procedure makes use of 190 different POS tags. The LSTM model is a sort of neural network that is well-known for its efficiency in natural language processing tasks, whereas the HMM model is a statistical model that is frequently used for POS tagging. The outcomes revealed that in the test dataset, the LSTM model had a better accuracy of 97%.

In addition to the quantitative analysis, a user interface was developed for interactive usage of the corpus and visualization of the results of analysis. The interface allows users to search for words and phrases in the corpus, view their frequency and usage, and visualize the results using graphs and charts such as Dispersion plots and frequency distribution plot. The result files were created in different formats, including CSV, JSON, and XML, and optimal formats were selected for the front-end part.

Overall, this research provides valuable insights into the Azerbaijani language corpus and its linguistic characteristics. The study demonstrates the potential of corpus linguistics and computational methods for studying languages and their characteristics. The findings can be used in various applications, including natural language processing, machine learning, and text analysis. Further research can be conducted to expand the corpus and explore other linguistic aspects of the Azerbaijani language.

Table of Contents

1 INTRODUCTION 1

1.1 DEFINITION OF THE PROBLEM 1
1.2 OBJECTIVE OF THE STUDY 2
1.3 SIGNIFICANCE OF THE PROBLEM 3
1.4 ASSUMPTIONS AND LIMITATIONS 5

2 LITERATURE REVIEW 6

3. RESEARCH APPROACH OR METHODOLOGY 11

3.1 DATASET DESCRIPTION 12
3.2 CORPUS CLEANING 13
3.3 POS TAGGING 14
3.4 FEATURE EXTRACTION FROM TEXT 21
3.5 N-GRAMS 25

4. RESEARCH RESULTS AND ANALYSIS OF RESULTS 26

5. SUMMARY AND FUTURE WORK 29

6 REFERENCES 31

LIST OF FIGURES

No	Figure Caption	Page
1	Overview of the system	12
2	Stopwords	13
3	Hidden Markov Model	15
4	RNN architecture	18
5	Inputs for RNN POS tagging	18
6	CBOW model of Word2Vec	24
7	RNN POS tagging model training and validation accuracy	28
8	Lexical Dispersion plot	28
9	Concordance analysis page from UI of the corpus	29

LIST OF TABLES

No	Table Caption	Page
1	Some of the most famous corpora	6
2.	Dataset Description	12
3	Examples to Bigrams	27
4	Categories and ratios of the corpus	30

LIST OF ABBREVIATIONS

Abbreviation	Explanation
POS	Part of Speech
ML	Machine Learning
DL	Deep Learning
HMM	Hidden Markov Model
LSTM	Long Short Term Memory
UI	User Interface
COCA	Contemporary corpus of American English
CL	Corpus Linguistics
PMI	Pointwise Mutual Information
RNN	Recurrent Neural Network
BPTT	Back Propagation through time
TF-IDF	Term frequency Inverse document frequency

1 INTRODUCTION

Computational Corpus Linguistics is a rapidly evolving interdisciplinary field that applies computational methods and techniques to study language through analyzing large collections of texts, known as corpora. Due to the exponential rise of digital data over the past few decades and the resulting abundance of textual data that is now available for study, this discipline has grown in importance. In several domains, including linguistics, natural language processing, and machine learning, new insights and discoveries have been made as a consequence of the processing and analysis of this data using computer approaches.

The current work, which employs computational corpus linguistics, focuses on the quantitative and semantic analysis of an Azerbaijani language corpus in this setting. In order to conduct the study's many quantitative and semantic analyses, including frequency analysis, n-gram analysis, concordance analysis, and word semantic similarity using multiple metrics, it is necessary to gather textual data on the Azerbaijani language. Additionally, Hidden Markov Model (HMM) and Long Short-Term Memory (LSTM) models for Part-of-Speech (POS) tagging were built and tested.

This thesis' two key contributions complement one another. First off, it offers a thorough examination of the corpus of Azerbaijani utilizing computational corpus linguistics, which has never been done previously. Second, it offers a useful method for developing an interactive user interface for corpus analysis and outcomes display. The study also emphasizes the value of computational corpus linguistics as a potent tool for deriving insightful results from language analysis.

1.1 Definition of the Problem

The problem addressed in this thesis is the lack of comprehensive studies on the Azerbaijani language corpus and its linguistic characteristics. While there have been some studies on the Azerbaijani language, there is a need for a more systematic and thorough analysis of the language corpus using state of art computational methods. This has become increasingly practical recently as a result of the growing accessibility of texts in digital format, which is pushing corpus linguistics in the direction of a "big data" approach. As a result, the quantitative techniques used in the discipline are growing increasingly complex and diverse.The lack of such studies limits the potential of the Azerbaijani language in various applications, including natural language processing, machine learning, text generation, text analysis. Therefore, the thesis aims to address this problem by creating a comprehensive Azerbaijani language corpus and conducting various quantitative and semantic analyses on the corpus. The research also aims to provide useful deliveries for development of a user interface for interactive usage of the corpus and visualization of the results of analysis, which can be used by researchers, language learners, and others interested in the Azerbaijani language. By addressing this problem, the thesis contributes to the development of corpus linguistics and computational methods for studying languages and their characteristics.

The collecting and processing of a sizable quantity of textual material from diverse sources is required in order to create a comprehensive corpus for the Azerbaijani language, therefore it is not a simple operation. In addition, the corpus must be correctly annotated with details like Part-of-Speech tags, which are essential for carrying out precise linguistic analysis. The Azerbaijani language's potential is however constrained by the absence of such a corpus, which also restricts the creation of tools and software for natural language processing.

1.2 Objective of the Study

The objective of this study is to use cutting-edge computer methods in Data Analytics and Machine Learning to carry out a thorough quantitative and semantic analysis of an Azerbaijani language corpus. The purpose of the research is to uncover linguistic patterns and regularities, such as common phrases and syntactic structures, and to thoroughly examine the language's semantic properties. Computational linguistics and natural language processing depend heavily on machine learning (ML) and data analytics. When dealing with intricate language patterns and structures, which are frequently challenging to recognize and evaluate manually, ML algorithms are very helpful. Researchers can discover patterns and regularities in language use that might otherwise go unreported by utilizing the capability of ML. The analysis of the Azerbaijani language corpus in this work makes substantial use of machine learning and data analytics. To find patterns and regularities in language use, the research uses a variety of methodologies, including n-gram analysis, collocation analysis, and concordance analysis. These methods provide academics the ability to recognize common words and syntactic patterns as well as to look at how these patterns are applied in various settings.

The study also evaluates the efficacy of alternative computational algorithms for corpus analysis using two unique Part-of-Speech tagging models based on Hidden Markov Models (HMM) and Long Short-Term Memory (LSTM) models. The study's ultimate goal is to further our knowledge of the linguistic properties of the Azerbaijani language and to aid in the development of natural language processing tools and applications for the language.

To achieve these objectives, the study will first gather a sizable textual dataset in the Azerbaijani language and build a corpus. After that, the corpus will go through a number of preparation processes, including cleaning, tokenization and normalization, to make sure the data is in a format that can be used for analysis.

The study will next employ a variety of quantitative techniques to find patterns and regularities in the language use, including n-gram analysis, collocation analysis, and concordance analysis. These techniques will make it possible to recognize common words and syntactic patterns and analyze how these patterns are applied in various settings.

Finally, the study will assists in providing deliveries for development of a user interface (UI) for interactive usage of the corpus and the visualization of the results of the analysis. The UI will enable researchers to interact with the corpus and to explore its linguistic features and characteristics in a user-friendly way. The study will also produce result files in different formats and select the optimal formats for the front-end part, making the corpus and its analysis accessible to a wider audience. Architecture of such a system for optimal user experience will be proposed as well. Architectural design of a language corpus system will include every component from Data warehouse for storage of our corpus and analytical results of that corpus to Database for fast retrieval of results for users and UI itself. User interface of the developed system send queries of the users to the database where results of analytics are stored. Queries could have different forms and the query syntax provides for a wide range of searches, including words, phrases, substrings, parts of speech, lemma, collocates, synonyms, bespoke word lists, limits by categories, sub-categories, and combinations of these.

The most basic type of search is probably finding the overall frequency of a word, phrase, substring, or grammatical structure across the six main genres (spoken, fiction, literary, newspapers, academic journals, and online). The results can also be sorted by word, lemma, or part of speech. Thesaurus and customized wordlists are another functionality present in the UI. In a typical thesaurus user finds synonyms of words. In the representation, one word will have many synonyms but of course, usage frequencies of these words will not be the same and

everyone will have different frequencies in different categories. In the development of thesaurus, we will take the advantage of relational databases allowing to add any number of additional tables with annotation features.

In order to implement the functionalities described above many quantitative methods have been executed. How statistics is used in quantitative corpus analysis and background knowledge for these methods are explained in this paper.

1.3 Significance of the Problem

Great portion of data available right now are in the form of text. It's a truism that 80 percent of business-relevant information originates in unstructured form, primarily text. The figure is very widely cited by analysts, vendors, and users alike, all seeking to make the case for text analytics. There are variations; Anant Jhingran of IBM Research, among others, cites an 85% figure [1]. But textual data is not structured and methods for mining valuable information from them are not clearly defined in contrast to the structured data. Furthermore, textual data contains morphological and syntactic structure and properties of a language it is written in. This feature makes the rule-based analysis of text even difficult.

A corpus is a group of text or audio that is composed of datasets and is used by native speakers and authors. It is a representation of language. A corpus could include anything from novels newspapers, posts, and comments in social networks to movies, radio and television show scripts. Corpus approach in language analysis has recently become more popular with the advances in computing power of computers. Primary goal of corpus-based analysis is to detect and interpret generalizable patterns in the use of language. Statistical analysis of the text, or in another word, corpus linguistics is more quantitative area of research and generally is about distribution of language token without consideration of semantic relationships among them. Corpus linguistics deals with wide varieties of frequency distribution and of tokens and probabilities of terms for the specific context in the corpus. It should be emphasized that corpus linguistics has recently begun to go beyond its restrictions of being a purely quantitative sub-discipline because to the availability of extra-large corpora for many languages, specialized platforms/tools, and computing capacity. Analysis of Azerbaijani corpus is designed to be utilized in text analysis or text mining applications.

System that is implemented as a presentation of corpus linguistics will consist of two components- quantitative analyzer of corpus and development of UI. In the first component huge corpus with tens of millions of words shall be processed to get analytical results, and the UI component shall present results of quantitative analysis. Details of functionalities of UI are specified in the sections below. Moreover, quantitative analysis methods applied to the corpus will be listed with the comprehensive background knowledge in the next sections.

The developed system will serve to the language analyzers to get some interesting insights and patterns of Azerbaijani language. System could be adapted to analyze any language given comprehensive corpus in that language, but system given in this paper shall work for Azerbaijani language.

Users of the system will interact with UI in order to get results of corpus analytics. System shall authenticate users that access the UI. System will also provide users with the following functionalities:

- Search

- Keyword in the Context
- POS list
- Matching forms
- Grouping by Lemma
- Token Collocates
- Word comparisons
- Thesaurus and customized wordlists
- Contrasting sections of the corpus
- Ngrams

Corpus contains more than 60 millions of words in total. Sub-categories of the corpus enable us to categorize topics of corpus more granularly. For example, spoken language category contains TV program scripts, theatre scenarios, scripts of series, radio program scripts, translations of foreign movie scripts. News section contains news from politics, economy, culture, religion, sport, science. Academic category contains subcategories like technology, science, medicine, law/politics, philosophy, history, geography, education etc. Web section has subcategories social platforms, blogs, Wikipedia data. As parts of the corpus contains different areas, they have different sources. Information about sources of data will be stored as a metadata in the database. Storing such metadata could be very useful for many purposes. One use case of metadata is the usage year of the text which could enable us to analyze how the language has changed throughout the years. Addition of texts to the corpus is expected to be two times in a year, and such analysis is not possible for now as our corpus encapsulates very limited time period for now. But over the years lexical, morphological, syntactic, semantic changes of the language could be analyzed by this data.

This work intends to address this gap and make a contribution to the creation of natural language processing tools and applications for the Azerbaijani language by undertaking a complete quantitative and semantic analysis of an Azerbaijani language corpus. The project will give important insights into the syntax, vocabulary, and semantics of the Azerbaijani language and make this knowledge available to a larger audience. The study's findings will be important to a range of stakeholders, including linguists, language teachers, software developers, and governmental agencies. A deeper comprehension of the language's structure and usage patterns will be advantageous for linguists and language educators since it can influence the materials used in language teaching and learning. The study's conclusions can be used by programmers to create natural language processing tools and apps for Azerbaijani speakers. The findings of the study can also be used by government agencies to assist plans for language planning and policy as well as to advertise the diversity of Azerbaijan's cultures and languages.

Overall, the importance of this work lies in its ability to aid in the creation of linguistic materials and tools for the Azerbaijani language's natural language processing and, in doing so, to promote a better knowledge and appreciation of this significant language and culture.

1.4 Assumptions and Limitations

It's critical to recognize and accept the study's limitations and underlying presuppositions in every research investigation. There is no exception to the rule in this work on the quantitative and semantic analysis of an Azerbaijani language corpus. The following are some of the study's limitations and assumptions.

The corpus utilized in this study was compiled from internet sources, therefore it might not be entirely representative of how the Azerbaijani language is used throughout the country. The corpus, for instance, can be skewed toward written language and fail to capture the subtleties of spoken language. As far as we tried to make unbiased corpus representing all aspects of Azerbaijani language, it is not possible to get completely balanced corpus due to the limitations in reaching every community that use Azerbaijani language.

Another limitation behind this work is that the Azerbaijani language POS tagging models currently in use have flaws that prohibit them from being very accurate. Particularly, the conventional Hidden Markov Model (HMM) and other machine learning methods employed for POS tagging in Azerbaijani may be unable to reflect the subtleties and complexity of the language. The paper suggests to extend size of POS tagged dataset to train models with greater data to create a more precise POS tagging model in order to overcome these constraints. Part-of-speech tagging is only one of several natural language processing tasks where deep learning models have made considerable strides in recent years. The suggested approach seeks to increase the precision of POS tagging in Azerbaijani by utilizing deep learning models like Long Short-Term Memory (LSTM) networks. Deep learning techniques have the ability to go beyond the limitations of traditional machine learning models by allowing the model to find intricate patterns and correlations within the language data. These methods can also record contextual data, such as word order and sentence structure, which are essential for correctly identifying a word's part of speech.

The suggested approach has the potential to offer superior POS tagging for the Azerbaijani language in terms of accuracy and dependability. If the model is effective, it may be used for a variety of natural language processing tasks, including sentiment analysis, named entity identification, and machine translation, which might have an impact on e-commerce, banking, and the healthcare sectors.

The study makes the assumption that the corpus's quantitative and semantic analysis can shed light on the Azerbaijani language's use and grammatical structure. But languages are dynamic systems that change through time rather than static items. Language change may be seen in phonology, morphology, syntax, and vocabulary, among other areas of language. Numerous things, including social, cultural, and historical forces, might cause these shifts. It is conceivable that the Azerbaijani language has changed over the years depending on the circumstance.

Another aspect of language change that can be observed in the Azerbaijani language is the evolution of grammar and syntax. Like other languages, Azerbaijani has undergone changes in its grammar and syntax over time. For instance, older forms of Azerbaijani may have had different word orders or grammatical sctures than cotruntemporary Azerbaijani. So, it is not exception that the Azerbaijani language will change in the future and this corpus could be outdated. This factor could be mitigated by continuously augmenting the corpus with recent examples of the language, which will also be examined in this study.

2 LITERATURE REVIEW

Literature review of this paper is divided into two parts. The first part is dedicated to various corpora developed in different languages and their architecture. In the second part, I will list the papers in which researchers mentions various statistical and semantic methods they have implemented for computational linguistics.

Numerous corpora have been created over time for several languages, advancing computational linguistics and natural language processing. With the use of these corpora, researchers have been able to explore a range of linguistic phenomena, including syntax, morphology, and semantics, as well as create models and algorithms for various natural language processing tasks. Some of the most famous corpora are given in the Table 2.

Table 2: Some of the most famous corpora

	Corpus name	**Docs**	**Words (mln)**	**Created**
1	Corpus of Contemporary American English (COCA)	485202	560	1990 - Now
2	American National Corpus	13295	22	1995 - 2005
3	Brown Corpus	500	1	1961
4	British National Corpus	4049	100	1991-1994
5	RC-Acquis (23 lang)	463792	636	1950-now
6	Russian National Corpus	4500000	1500	2003-2005
7	Turkish National Corpus	4990	50	1990-2013

In his paper [9], Mark Davies of Brigham Young University talks about a corpus that he designed. This corpus is made up of over 385 million words from the period of 1990-2008 and has a balance of materials from spoken sources, fiction, popular magazines, newspapers, and academic journals. Additionally, he highlights the unique relational database architecture that was used to create this corpus, which enables a wide range of queries that cannot be performed with other interfaces and architectures. In terms of text kinds, the corpus was created to be roughly equivalent to the BNC. About 10% of the texts in the BNC are spoken, 16% are fiction, 15% are from (popular) periodicals, 10% are from newspapers, and 15% are scholarly; the other texts are from various genres. The percentage of spoken (20%), fiction (20%), popular magazines (20%), newspapers (20%), and scholarly journals (20%) texts in the COCA is distributed equally. This composition is true for the corpus as a whole as well as for each individual year. As a result, researchers may compare data asynchronously throughout the corpus and be pretty certain that the composition of an analogous text from year to year will appropriately reflect linguistic changes. He says that the volume of content that is already in electronic format and available over the Web makes building a corpus in 2008 - as opposed to 15-20 years ago, when the BNC was being developed - obviously advantageous.

These corpora's design is built on a heavy reliance on relational databases. The core database has a table with 385+ million rows for a corpus with over 385 million words, one row for each token

in sequential order. A [wordID] column in the table includes the integer value for each distinct type in the corpus (wordID), and a [textID] number in the table corresponds to one of the more than 150,000 texts in the corpus. The [ID] column displays the sequential position of each word in the corpus (1, 2, 3,... 385,000,000). The [wordID] value in the 'dictionary' table corresponds to the [wordID] value in the previous table and provides part of speech, lemma, and frequency data for each of the 2.3 million types in the corpus. Each of the more than 150,000 texts in the corpus has metadata in the'sources' table, including details on the genre, subgenre, title, author, and source (such as the magazine, issue, and pages). There are further tables with additional word-level data as well. These include tables for user-created wordlists and a "synonyms" table with more than 370,000 items (synonyms for more than 30,000 words). According to his assessment, the relational database architecture provides various notable benefits over rival architectures. The foremost advantage is related to its speed and size, as each table, including the usage of clustered indexes, is indexed, resulting in swift query processing even for large corpora. Another major advantage of the relational database architecture is its ability to support a modular structure that permits the incorporation of numerous additional feature sets into the architecture without compromising its speed.

Another research was made by V´ıt Baisa and V´ıt Suchomel about Large Corpora for Turkic Languages and Unsupervised Morphological Analysis [13]. This article introduces six brand-new online corpora for the languages of Turkish, Azerbaijani, Kazakh, Turkmen, Kyrgyz, and Uzbek. SpiderLing used the web to autonomously crawl the data for these corpora. To get the data in its raw state, just a very basic understanding of these languages was needed. SpiderLing is web crawler and is made to collect linguistic information from the internet for research needs. It may be used to gather text, audio, image, and other sorts of data from numerous online sources, including social networking sites, blogs, and news websites. To learn more about language usage, trends, and patterns, SpiderLing's data may be studied. Them mentions that Turkic languages are interesting because they feature a huge variety of word forms due to their productive inflectional and derivational agglutinative morphology. If Turkish and English corpora of same size are compared, the latter will include far more wordforms but at lower frequency. Large corpora are thus considerably more essential for these languages.

In the research called How to use statistics in quantitative corpus analysis Stefan Th. Gries and Justus Liebig say that typically, corpora do not offer direct insights into the aspects of language that most linguists are interested in, such as meaning, communicative function, information structure, cognition, language proficiency, or dialect. Instead, a typical corpus is limited to providing information on the occurrence or non-occurrence of character strings, usually in the form of graphemes like letters from any language. There are two implications of this limitation. Firstly, any linguistic inquiry based on corpus data must rely on the frequency of occurrence of a particular text or annotation within the corpus, or the co-occurrence of two or more linguistic features. The second implication is that the concept of correlation will eventually become a crucial tool in corpus-linguistic analysis. In order to study meaning, such as the semantics of a specific word or argument structure construction, a linguist must retrieve examples of the word or construction from the corpus, annotate them for characteristics of interest, and then correlate those characteristics with each other and/or with the word or construction's meanings or types of usage. Similarly, to investigate the function of information structure in constituent ordering, one

must obtain examples of the relevant constructions from the corpus, annotate them for information-structural and other features, and then establish correlations between these features and the constructional choices made. Despite seeming insignificant, the importance of correlation means that corpus linguistics is often closely associated with statistical analysis, as

statistical methods are used to interpret frequencies, distributions, and correlations. They also mention the majority of linguists are often interested in meaning, communicative function/intention, information structure, cognition/processing, and language proficiency/dialect, which corpora typically do not directly give. Instead, a prototype corpus reports whether or not the specific character strings—typically a grapheme like a letter (in any language), a space, a number, or special character—are present in the corpus or in a particular context.

Correlation is a concept that is involved in corpus-linguistic analysis. One must typically retrieve examples of the word or construction from one corpus, annotate them for features of interest, and then correlate those characteristics with one another and/or with the meanings or types of uses of the word or construction in order to study meaning, for example, the semantics of a particular word or argument structure construction. Similar to this, to study information structure and its function in constituent ordering, one must typically retrieve examples of the constructions in question, annotate them for information-structural and other characteristics, and then correlate those with one another and the constructional decisions. As unimportant as that may appear, the fact that corpus linguistics frequently has close ties to statistical analysis shows how dependent it is on the concept of correlation and how we interpret correlations, frequencies, and distributions.

Token frequency of occurrence, or the notion of how frequently a single word, n-gram, a grammatical construction, etc. exists in a corpus or in a specific part of a corpus, is the most fundamental type of statistic. From the frequency statistics statements like word a occurs n times in the corpus could be inferred. Such kind of inference could be contrasted with another one for the word b. Nevertheless, we frequently need to compare several corpora that are not all of the same size. In these cases, normalizations like per thousand words, per million words are common. And alike inferences can be made for term frequencies of co-occurrences.

Due to these term frequencies' relationships with numerous experimental tasks (word naming, retrieval of a word), which have been significant in many various fields, frequency is regularly used as an explanatory or merely a control variable in corpus analysis.

Type frequency, or the number of types that, for example, occur in a specific semantically or syntactically specified location, is another type of frequency information. For instance, a corpus may contain 100 cases of the noun lemma followed by a nominal direct verb. Those 100 tokens may represent 40 different noun lemma types, including some that are extremely common, some that are moderately frequent (which may represent another 20 tokens), and some that are extremely uncommon.

Type frequencies have been thought to be crucial for studies of productivity, such as when a productive morpheme is attached to more types than an unproductive one, or studies of category formation, such as when a semantical item turns into a grammatical item by virtue of being associated with many semantically extremely diverse words. Phraseologisms or static phrases may be detected by very low type frequency as well: This is a good indication that it could be a static expression because the word that follows the adverb tightly occurs with a type frequency of 1 in almost all corpus (because sealed is almost always the following word).

Frequency data are among the most often offered corpus statistics because of their importance and, in particular, the relative ease with which they may be produced. However, there are certain

conceptual and methodical issues with frequency. When it comes to language and cognition, for example, frequency is connected with various other features, but this does not always mean that it also has a potential influence on these other aspects, which is a completely separate hypothesis to prove or disprove.

Yet, publishing a frequency alone, especially a term frequency, is as problematic from a quantitative and methodological standpoint as reporting an average (mean) alone. For that reason, corpus frequencies as also stated in statistical adage- mean value must never be expressed alone without giving dispersion applies to term frequency as well. That is why, in the next section importance of dispersion and how they could be used for analysis of language corpus have been stated.

The term "dispersion" in corpus linguistics refers to how evenly an element is spread across a corpus, which is similar to the term "dispersion" in statistics. That definition also implies that the concept of recency and dispersion in corpus linguistics are connected.

For example, word a and b could have same frequency in the corpus but a is mostly placed in the specific context, but word b is spread across the corpus. So, if one part of the corpus is selected randomly there is more change that it contains the word b rather than word b.

Typically, frequency and dispersion are connected. It is obvious that, high-frequency function words will be distributed most evenly, and naturally, words with a frequency of 1 will be distributed most unevenly. Despite this overall relationship, dispersion and frequency diverge most in the range of moderately frequent lexical words, which is why it is important to always keep them apart. As a result, dispersion should be a key corpus statistic for any corpus based linguistic applications. Additionally, preliminary results show that dispersion can now beat frequency as a predictor of the changes in a language. Basically, frequency is usually the simplest statistic to collect but should always be supplemented with a dispersion measure if a linguist wants to analyze the likelihood that a linguistic element is recognized by the user or the familiarity of a linguistic element to the user.

Another important fundamental statistic given in this paper, known as cooccurrence statistics, includes a little amount of contextual data. Co-occurrence matrices could be used for various purposes regarding language analysis. In a co-occurrence matrix, distinct entities will typically be found in the rows and columns of the matrix. This matrix serves the goal of listing the frequency with which each row entity and each column entity coexist. As a result, you must define your entities and the context in which they co-occur in order to apply a co-occurrence matrix. The most traditional method of NLP is to define each entity (such as lines and columns) as a word that can be found in a text and the context of the word as a sentence. Co-occurrence matrices could be first order, second order and so one. Numbers here represents the size of sliding window. For example, first order only takes words that locates directly before or after the word in context.

One use case of second order co-occurrence matrix is by using pointwise mutual information (PMI) value determining semantic similarity of target words. Related work in that context has been carried out and results on synonym tests of English language exam show that this approach outperform other approaches. Pointwise mutual information (PMI) score is calculated using type frequency function, bi-gram frequency function and vocabulary of corpus. Information about type frequency is given above while talking about term frequencies and detailed information of calculating pointwise mutual information value will be given in the preceding sections of this paper [5].

Second Order Co-occurrence PMI for Determining the Semantic Similarity of Words, an article by the other authors, describes a novel corpus-based method for determining the semantic similarity of two words. The SOC-PMI approach uses Pointwise Mutual Information to determine each target word's important neighbors before finding the terms that are shared by both lists to determine their relative semantic similarity. These frequent terms from the opposing list's PMI values are combined for this use. The authors have established a semantic similarity function based on PMI, which compares two given words, W1 and W2.

(2.1) $$Sim(W_1, W_2) = \frac{f^{\beta}(W_1)}{\beta_1} + \frac{f^{\beta}(W_2)}{\beta_2}$$

Where f(W) is β-PMI summation function and defined by

(2.2) $$f^{\beta}(W_1) = \sum_{i=1}^{\beta_1} \left(f^{pmi}(X_i, W_2) \right)^{\gamma}$$

And betta -depend on the word W and the number of types in the corpus. Also fpmi() defined by:

(2.3) $$f^{pmi}(t_i, W) = \log_2 \frac{f^{b}(t_i, W) \times m}{f^{t}(t_i) f^{t}(W)}$$

Where m is total number of tokens in corpus C, f t (ti) tells us how many times the type ti appeared in the entire corpus and fb(ti,W) is bigram frequency function.

Co-occurrence of Second Order The same authors' publication [6], PMI for Determining the Semantic Similarity of Words, provides a brand-new corpus-based technique for assessing the semantic similarity of two words. The SOC-PMI technique finds the terms that are shared by both lists to ascertain the relative semantic similarity of each target word's neighbors using Pointwise Mutual Information. The PMI values of these common words from the opposite list are pooled here.

The Zipfian Approach to Words in Context has been used in yet another intriguing research.[7] The inverse link between a word's frequency in the corpus and its rank in the frequency table is stated by Zip's law using quantitative statistics. As a result, the most frequent word will be used around twice as frequently as the second-most frequent word, three times as often as the third-most frequent phrase, etc. The application of Zipf's law to the field of linguistics is expanded in this work. Instead of viewing entire texts as collections of discrete linguistic units, we have reduced Zipfian linguistics to a relatively small scale by treating the contextual words as connected by their syntactic links with the word in question. Investigating the systemic behavior of particular linguistic units is another advantage of utilizing a words-in-contexts paradigm.

These investigations' findings have also helped connect important facets of human language to Zipf's law's forms. For instance, it has been found that the Zipfian line's steepness is connected with lexical variety and degree of syntheticity, two characteristics that are closely related to language.

Words are an example that may be used. When a word is really employed, it is located in the context of other words that are connected to it. This is a language sub-system that involves both syntagmatic and paradigmatic links. Is it possible to analyze the circumstances of specific words in a similar way to how a Zipfian technique may be used to investigate language as a system? If so, how might the Zipfian features of the context describe the target words' usage patterns? This work seeks to respond to the problems posed above by analyzing the link between the Zipfian features of word contexts (as language subsystems) and the classes of the words in question.

In another similar article authors write about empirical phenomena in word frequencies.[8] They mention that the properties of word frequencies reviewed in the method-The general method followed in this article is to study relevant subsets of the lexicon and quantify the fit of Eq. 2-

will have much to say about the most plausible accounts of the word frequency distribution in general.

$$f(r) \propto \frac{1}{(r+\beta)^{\alpha}} \tag{2.4}$$

In this equation, r stands for a word's frequency rank, and f(r) stands for the frequency of that word in a natural corpus. This law states frequencies proportionally: The most frequent word (r = 1) has a frequency proportional to 1, the second most frequent word (r = 2) has a frequency proportional to, the third most frequent word has a frequency proportional to, and so on. The actual observed frequency will depend on the size of the corpus being studied. And $\alpha \approx 1$ and $\beta \approx 2.7$.

It certainly appears like we use words to express intended meaning as language users. From this straightforward vantage point, Zipf's rule essentially describes the "need" distribution for how frequently we must convey each meaning. Surprisingly, many explanations of the law derive it from principles unrelated to the substance of language, omitting any mention of meaning and semantics. But this viewpoint is incongruous with the fact that meaning and frequency are systematically associated, even across languages. Calude and Pagel (2011) compared the word frequencies on Swadesh lists from 17 languages from six language groups. Swadesh lists offer translations of common, everyday terms like "mother" into several languages; they are frequently employed in historical research. For these frequent terms, Calude and Pagel found an average interlanguage correlation in log frequency of R^2 =.53 (p .0001), showing that word frequencies are unexpectedly resilient across languages and predicted from their meanings. The predicted R^2 is almost certainly lower for less common words since Swadesh words have a tendency to occur often. In any event, a convincing explanation of the frequency distribution must take into consideration meaning if it affects frequency in any way.

The frequency-rank plots of the Swadesh lists published by Calude and Pagel (2011) demonstrate the systematic frequency-rank connection that exists across languages. These diagrams demonstrate how frequently each word is used in various languages and categorize terms according to their meanings. Using a fixed frequency rank that was separately assessed on 25% of the data from each language and then compressed across languages, the frequency-rank charts in this article were created. This method enables meaningful comparisons of word frequency between languages and helps adjust for variations in vocabulary size among languages. Overall, these stories offer insightful perspectives on the parallels and discrepancies across languages, and they may be utilized to learn more about the structure and historical development of language. This implies that the Swadesh list's order of words, denoted by their x-location, is constant across all languages and is only based on the total frequency of those words' use across all languages, denoted by their y-location. If words follow similar patterns across languages, it may be determined by comparing the frequency of terms at each rank. The Swadesh lists' frequency-rank plots show that word frequency patterns are remarkably constant across languages and, when the ranks are combined, exhibit a virtually Zipfian distribution.

3. RESEARCH APPROACH OR METHODOLOGY

The system that implements quantitative analysis of language corpus consists of the components described in the Figure 1. Data warehouse will be located in the server and will be used as a repository for language corpus. We need to store and maintain the corpus even after the

quantitative analysis ends because as stated in the requirements documents corpus will be updated and extended from time to time and analysis is executed in the corpus not in a part of corpus. Corpus analysis script will work as a pipeline to transform corpus data from input form to output form and then will execute some statistical analysis methods on it. Produced results from the analysis will be stored in the database server and backend server of web application will query from the database in request of user. The system will enable its users to interact with it via UI. Data warehouse, database and backend servers could be located in the same machine.

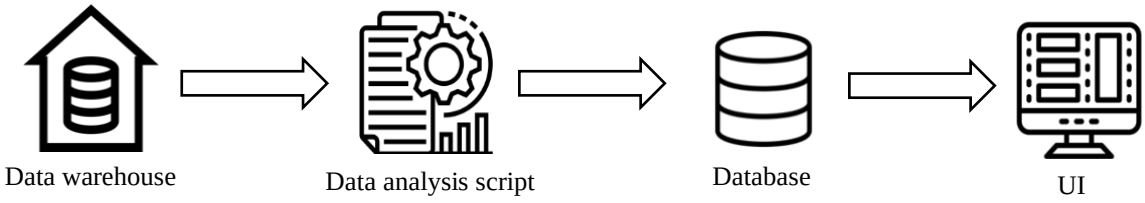

Figure 1. Overview of the system

Some background on quantitative corpus analysis. Corpus is simply text files from various sources and about different topics. Corpus will also have metadata about source of text and its category (politics, economy, education etc.). This metadata will be useful for analyzing which words are mostly used in what content. The next subsections of this chapter include main approaches and functions used in the paper for quantitative and semantic analysis.

3.1 Dataset Description

The dataset utilized in this thesis is a collection of texts written in Azerbaijani that were gathered from a variety of news sources. The dataset, which contains almost 60 million words, is a sizable and accurate sample of the Azerbaijani language.

A wide range of subjects are covered by the texts in the collection, including politics, economics, sports, culture, and entertainment. The dataset contains texts of varying lengths, ranging from brief news pieces to extensive literary works, offering a wide spectrum of linguistic complexity and use. General description of used dataset is given in the table below:

Table 2: Dataset Description

Category	**Ratio**
Size	1.2 GB
Sentences	3,492,453
Tokens	59,666,273
Types	517,829
Stop words used	204

The dataset underwent preprocessing to exclude any noise or unimportant data, including HTML elements, punctuation, and stop words. To decrease the dimensionality of the dataset, the

texts were then tokenized into words and phrases, with the words being lemmatized to their simplest forms.

To train and assess machine learning models, such as the POS tagging models, the dataset was divided into training, validation, and test sets. The validation and test sets each make up 10% of the dataset, whereas the training set makes up 80% of it.

In order to compare the effectiveness and accuracy of the two methods, HMM and LSTM models, the dataset was annotated with POS tags. Numerous quantitative and semantic studies, including frequency analysis, n-gram analysis, and concordance analysis, were also performed on the annotated dataset. Overall, the dataset employed in this thesis represents a representative and varied sample of the Azerbaijani language, allowing the researcher to apply computational corpus linguistics approaches to carry out a thorough examination of the language.

3.2 Corpus cleaning

Natural language processing (NLP) requires that a corpus, or huge collection of textual data, be cleaned in order to eliminate any noise or extraneous information. This procedure is required to guarantee the accuracy and dependability of the data utilized in the analysis. Depending on the kind of data and the particular requirements of the study, many strategies can be employed for corpus cleaning. Getting rid of any undesirable characters or symbols from the corpus is the first step in corpus cleaning. This can include, among other things, punctuation, special characters, and HTML elements. These components might bring noise into the data, which makes it more challenging to extract useful information, and are frequently unrelated to the analysis.

The corpus must next be cleaned up of any duplicate or nearly identical documents. Numerous methods, including similarity metrics and clustering algorithms, can be used to accomplish this. Duplicate removal reduces the corpus size and increases the effectiveness of future analysis. Eliminating stop words—common terms that have little significance in the text is another crucial step in corpus cleaning. Total number of stop words used were 203. By putting more emphasis on words with more significance, eliminating stop words can shrink the corpus size while also increasing the precision of subsequent analysis.

These steps are vital as they have significant effect on the final results of any model.

a	barı	çünki	fəqət	mı
afərin	başqa	da	güman	nəhayət
aman	bax	daha	halbuki	nəinki
amma	belə	demək	haqqında	ötrü
ancaq	bəlkə	deməli	hər	qeyri
arasında	bəs	doğrusu	hətta	sarı
artıq	bıy	ə	ki	sonra
axı	bu	əgər	lakin	tək
ayrı	bütün	elə	lap	üçün
az	çox	ən	məhz	və

Figure 2 Stop words

Some of the used stopwords are given in the Figure 1.

3.3 POS tagging

Grammatical tagging, commonly referred to as part-of-speech (POS) tagging, is the process of giving words in a sentence tag based on their syntactic and morphological characteristics.
These tags identify each word in the phrase according to its function, including its noun, verb, adjective, adverb, preposition, conjunction, and interjection.
Since it forms the basis for many subsequent tasks including information extraction, sentiment analysis, and machine translation, POS tagging is an important piece of work in computational linguistics and natural language processing. In addition to requiring a full understanding of the language's grammar and syntax as well as the context and meaning of the words, POS tagging is a challenging task. POS tagging is used in several industries, such as education, information retrieval, and natural language processing. The importance and breadth of this task for the advancement of natural language processing have lately been highlighted by the state-of-the-art performance in POS tagging achieved by Long Short-Term Memory (LSTM) networks and Transformer models, two deep learning-based methodologies. For POS tagging, a variety of methods can be employed, including:

- Rule based tagging- The rule-based POS tagging models give POS tags to words by applying a set of handwritten rules and making use of contextual information.
- Transformation based tagging- The transformation-based approaches combine a set of predetermined rules that are manually established with a collection of rules that are automatically formed during the training process.
- Deep learning models- Deep learning models, including Meta-BiLSTM, have been utilized for POS tagging and have shown impressive accuracy, with Meta-BiLSTM achieving an accuracy of nearly 98 percent.
- Probabilistic tagging- The best tag for a word is chosen using statistics, probability, or frequency in a stochastic method to POS tagging. To tag a word in unannotated text, one straightforward method is to find the most frequently used tag for that word in the training data. This strategy, meanwhile, occasionally leads to tags for sentences that do not follow a language's grammatical norms. Choosing the tag sequence with the highest probability necessitates using a different strategy that involves assessing the probabilities of various tag sequences that may be utilized for a particular text. For POS tagging, one example of a probabilistic method is the Hidden Markov Model (HMM).

Two methods are considered in POS tagging for comparison purposes in this paper. Rather simple Machine Learning model uses Hidden Markov Model for part of speech tagging and the main model that is more complex uses Deep Learning techniques, more specifically Recurrent Neural Network.
Tokenization, stemming, lemmatization, parsing, and part-of-speech (POS) tagging are just a few of the natural language processing (NLP) operations that the Natural Language Toolkit (NLTK), a well-known open-source Python toolkit, can do. For a variety of NLP applications, including sentiment analysis, text classification, and information retrieval, the NLTK POS tagger offers a means to automatically annotate text with information about the parts of speech of words in a sentence. The Hidden Markov Model (HMM), a statistical model that may be used to forecast the sequence of POS tags for a given phrase based on the likelihood of detecting a certain

tag given the previous tag and the current word, is the foundation of the NLTK HMM POS tagger.

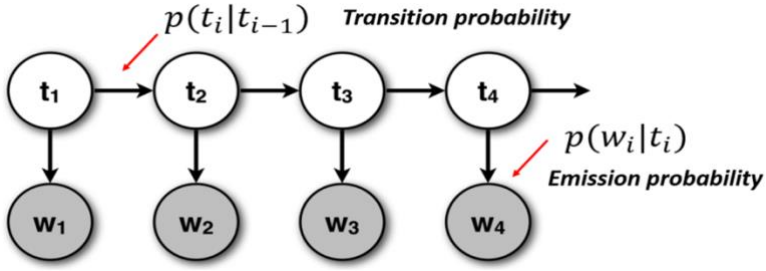

Figure 3: Hidden Markov Model

Being a generative model, the HMM model describes both the hidden states (POS tags) and the probability distribution of the observations (words) that produce them. We need to know the Transition likelihood and Emission Probability of each tag in order to determine the likelihood attached to this particular sequence of tags.

The possibility of a specific sequence, such as whether a noun would be followed by a model, a model by a verb, and a verb by a noun, is measured by the transition probability. The Transition Probability is the name given to this likelihood. For a certain sequence to be accurate, it should be high.What is the likelihood that the words Ted, Will, Spot, and Will are nouns as opposed to verbs? These probability sets, which are known as emission probabilities, must be large in order for our tagging to be plausible.

The HMM model is trained using a labeled corpus of text, where each word is assigned its matching POS tag, throughout the training process. A training set and a test set are often created from the corpus, with the training set being used to estimate the model's parameters (transition and emission probabilities) and the test set being used to assess the model's correctness. The HMM model calculates the probability of transitioning between various tags and the odds of emitting a certain phrase given a specific tag. While the emission probabilities describe the likelihood of seeing a certain phrase given a specific tag, the transition probabilities represent the likelihood of migrating from one tag to another. Using maximum likelihood estimation or other approaches, these probabilities are calculated from the training corpus. The most likely sequence of POS tags for a new phrase may be predicted using the trained model. First, a collection of potential tag sequences for the supplied phrase are generated by the HMM POS tagger using the transition probabilities that were discovered during training. Using the emission probabilities, it then determines the likelihood of each potential sequence and chooses the most probable sequence to serve as the sentence's final POS tags. On common benchmark datasets, the NLTK HMM POS tagger may attain accuracy levels of 90–95% with respect to speed. It cannot, however, capture long-range connections between words in a phrase, and addressing uncommon or obscure terms might be challenging. Researchers have suggested a number of additions and changes to the fundamental HMM model to solve these constraints. Using extra features (such word context, syntactic patterns, and lexical resources) to increase the tagger's accuracy is one method, known as feature-based tagging. Another strategy is to employ deep learning models like recurrent neural networks (RNNs) or transformers, which are better equipped to handle unknown words than HMMs and can capture long-range relationships.

A Hidden Markov Model is employed by the HMM-based POS tagger in NLTK to forecast the POS tags. The observation model and the state transition model are the two halves of the HMM.

The observation model determines, given a certain POS tag, the likelihood of witnessing a word. The chance of switching from one POS tag to another is determined by the state transition model. It uses a corpus of annotated text to initially train the model. The tagger gains knowledge of the odds of witnessing each word given a certain POS tag as well as the probabilities of switching between POS tags during training. The POS tags for unseen text may be predicted using the model after it has been trained. The HMM-based POS tagger first creates a list of potential POS tags for each word in the sentence before tagging it. It then uses the state transition model and the observation model to determine the likelihood of each potential tag sequence. The most likely tag combination is then chosen as the appropriate POS tags for the statement. The HMM-based POS tagger's ability to handle unseen words is one of its benefits. The tagger can utilize the observation model to calculate the likelihood of each potential tag for a word based on its context if a term not in the training data is encountered. As a result, the tagger is more adaptable to new terms and may be used on a variety of texts. The HMM-based POS tagger's ability to be trained

on tiny datasets is another benefit. For languages like Azerbaijani, where there may not be a sizable corpus of annotated text accessible, this is very helpful. On unseen text, it is still feasible to get a respectable level of accuracy by training the tagger on a limited sample.

Code 1: HMM POS tagging

```
import nltk
from nltk import word_tokenize
nltk.download('averaged_perceptron_tagger')
nltk.download('webtext')
nltk.download('punkt')


with open('The_Last_One_Full/The_Last_One_Labels_POS_Tagging.txt', 'r', encoding='utf-8') as f:
    tags = f.read()
with open('The_Last_One_Full/The_Last_One_Sentences_POS_Tagging.txt', 'r', encoding='utf-8') as f:
    corpus = f.read()

sentences=corpus.split('\n')
sent_tags=tags.split("\n")
words=[ sent.split(' ') for sent in sentences]
final_tags=[tag.split(' ') for tag in sent_tags]
tagged_sentences=[]

for sent,tag_sent in zip(words,final_tags):
    tagged_sent=[]
    for w,t in zip(sent,tag_sent):
        tagged_sent.append((w,t))
    tagged_sentences.append(tagged_sent)

# Split the corpus into training and testing sets
train_sents = tagged_sentences[:int(len(tagged_sentences) * 0.8)]
test_sents = tagged_sentences[int(len(tagged_sentences) * 0.8):]


# Define the POS tagger model and train it
trainer = nltk.HiddenMarkovModelTrainer()
tagger = trainer.train_supervised(train_sents)
```

RNN POS tagger

When processing sequential data, such as time series or text written in natural language, recurrent neural networks (RNNs), a particular sort of neural network design, are very helpful. RNNs keep an internal state that enables them to analyze data of any length and take into consideration prior inputs while creating output, in contrast to standard feedforward neural networks, which process input data in a single pass and create output. RNNs can thus be seen to have a "memory" of prior inputs, allowing them to recognize temporal connections in the data. The recurrent neuron is the fundamental component of an RNN and it accepts as input both the current input data and the output from the previous time step. The neuron calculates an activation value for the subsequent time step depending on the input and the previous output at each time step. As a result, the output at each time step depends on the network's internal state as well as its prior inputs and current input. Similar to the backpropagation technique used to train feedforward neural networks, backpropagation through time (BPTT) may be used to train RNNs. In BPTT, the loss function's gradient with respect to the network's parameters is calculated at each time step, and the gradients are then used to update the parameters using an optimization approach like stochastic gradient descent (SGD). The vanishing gradient problem, which happens when the gradients get very tiny as they are transmitted backward in time, is one difficulty with training RNNs. Because information from earlier time steps could already be "forgotten" by the time it gets to later time steps, this might make it challenging for the network to understand long-term dependencies. The Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) designs, among others, have been suggested as adaptations to the fundamental RNN architecture to deal with this problem. In order to solve the vanishing gradient problem, gated recurrent neurons such as LSTMs and GRUs enable the network to selectively retain or forget data from earlier time steps. LSTMs do this by controlling the flow of information into and out of the cell state, which acts as the network's "memory," using three gating mechanisms: the input gate, the forget gate, and the output gate. Input and forget gates are combined into a single "update gate" in GRUs, and a separate "reset gate" is used to regulate the information flow from the previous output to the current input. RNNs have been effectively used for a variety of applications, including speech recognition, language modeling, machine translation, and picture captioning. One standout is Google's Neural Machine Translation system, which combines LSTMs and attention mechanisms in tandem to achieve cutting-edge performance on a variety of machine translation benchmarks. RNNs have been utilized in more contemporary applications, such as generative models, where they are used to create new text or other sorts of data based on precedent samples, in addition to their effectiveness in typical machine learning tasks. RNN-based models, for instance, have been used to create original poetry, music, and even computer code.

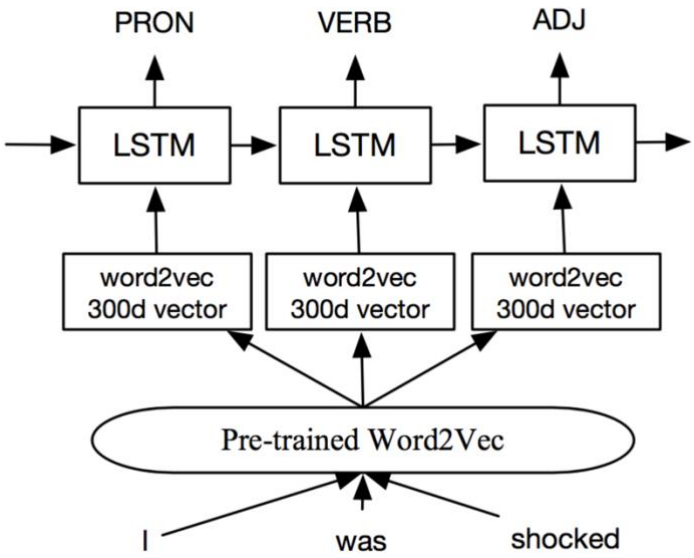

Figure4: RNN architecture

In a vanilla RNN (Recurrent Neural Network), the output of the previous time step is provided as input to the current time step, which is employed for sequential data processing. It is especially helpful in natural language processing (NLP) activities including machine translation, part-of-speech (POS) tagging, and language modeling. A hidden state vector and a set of weights that are used in every time step make up an RNN's basic architecture. The input and the previous hidden state vector are sent into the RNN at each time step. The hidden state vector, which is updated with the current input and prior hidden state, is the RNN's output. For POS tagging, a series of word embeddings—each word being represented as a high-dimensional vector—are used as the RNN's input. The RNN is fed the embeddings one at a time, and at each time step, its output is utilized to forecast the POS tag for the accompanying word. A labeled dataset of words and their related POS tags is utilized to train the vanilla RNN for POS tagging. Minimizing the cross-entropy loss between the anticipated POS tags and the ground truth tags is the objective. The RNN's weights are updated using the backpropagation technique to reduce loss.

```
1 tagged_sentences[0]
```

```
[('Aparılmış', 'Feil-Sifət'),
 ('təxirəsalınmaz', 'Sifət'),
 ('istintaq-əməliyyat', 'İsim'),
 ('tədbirləri', 'İsim'),
 ('nəticəsində', 'İsim'),
 ('Z.Ömərov', 'İsim-Mürəkkəb_Ad'),
 ('şübhəli', 'Sifət'),
 ('şəxs', 'İsim'),
 ('qismində', 'İsim'),
 ('tutularaq', 'Feil-Bağlama'),
 ('müdafiəçi', 'İsim'),
 ('ilə', 'Qoş-Vasitə_birgə'),
 ('təmin', 'B/Feil'),
 ('edilməklə', 'E/Feil'),
 ('dindirilib', 'Feil')]
```

Figure 5 :Inputs for RNN POS tagging

Code 2: collecting inputs for RNN model

```
X = [] # store input sequence
Y = [] # store output sequence
for sentence in tagged_sentences:
    X_sentence = []
    Y_sentence = []
    for entity in sentence:
        X_sentence.append(entity[0]) # entity[0] contains the word
        Y_sentence.append(entity[1]) # entity[1] contains corresponding tag

    X.append(X_sentence)
    Y.append(Y_sentence)
num_words = len(set([word.lower() for sentence in X for word in sentence]))
num_tags   = len(set([word.lower() for sentence in Y for word in sentence]))
print("Total number of tagged sentences: {}".format(len(X)))
print("Vocabulary size: {}".format(num_words))
print("Total number of tags: {}".format(num_tags))
```

```
Total number of tagged sentences: 30689
Vocabulary size: 56515
Total number of tags: 190
```

190 unique tags are used over 56515 unique words in vocabulary in training text for POS tagging model.

Code 3: Encoding inputs to numerical format

```
# encode X
from tensorflow.keras.preprocessing.text import Tokenizer
word_tokenizer = Tokenizer()              # instantiate tokeniser
word_tokenizer.fit_on_texts(X)            # fit tokeniser on data
# use the tokeniser to encode input sequence
X_encoded = word_tokenizer.texts_to_sequences(X)
# encode Y
tag_tokenizer = Tokenizer()
tag_tokenizer.fit_on_texts(Y)
Y_encoded = tag_tokenizer.texts_to_sequences(Y)
# look at first encoded data point
print("** Raw data point **", "\n", "-"*100, "\n")
print('X: ', X[0], '\n')
print('Y: ', Y[0], '\n')
print()
print("** Encoded data point **", "\n", "-"*100, "\n")
print('X: ', X_encoded[0], '\n')
print('Y: ', Y_encoded[0], '\n')
```

```
** Raw data point **
 ----------------------------------------------------------------------------------------------------

X:  ['Aparılmış', 'təxirəsalınmaz', 'istintaq-əməliyyat', 'tədbirləri', 'nəticəsində', 'Z.Ömərov', 'şübhəli', 'şəxs',
'qismində', 'tutularaq', 'müdafiəçi', 'ilə', 'təmin', 'edilməklə', 'dindirilib']

Y:  ['Feil-Sifət', 'Sifət', 'İsim', 'İsim', 'İsim', 'İsim-Mürəkkəb_Ad', 'Sifət', 'İsim', 'İsim', 'Feil-Bağlama', 'İsi
m', 'Qoş-Vasitə_birgə', 'B/Feil', 'E/Feil', 'Feil']


** Encoded data point **
 ----------------------------------------------------------------------------------------------------

X:  [4887, 7946, 15451, 819, 142, 23395, 3500, 643, 1590, 6896, 23396, 5, 143, 6083, 23397]

Y:  [13, 2, 1, 1, 1, 30, 2, 1, 1, 29, 1, 22, 4, 5, 3]
```

Code 4: Padding sequences for coherent format

```
from keras.utils import to_categorical
from tensorflow.keras.preprocessing.sequence import pad_sequences
MAX_SEQ_LENGTH = 100
X_padded = pad_sequences(X_encoded, maxlen=MAX_SEQ_LENGTH, padding="pre", truncating="post")
Y_padded = pad_sequences(Y_encoded, maxlen=MAX_SEQ_LENGTH, padding="pre", truncating="post")
# print the first sequence
print(X_padded[0], "\n"*3)
print(Y_padded[0])
```

```
[    0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0  4887  7946 15451   819   142 23395  3500   643  1590  6896 23396
     5   143  6083 23397]



[ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  0 13  2  1  1  1 30  2  1  1 29  1
 22  4  5  3]
```

Code 5: Structure of RNN model

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding
from tensorflow.keras.layers import SimpleRNN
from tensorflow.keras.layers import TimeDistributed
import tensorflow as tf

rnn_model = Sequential()
# create embedding layer — usually the first layer in text problems
# vocabulary size — number of unique words in data
rnn_model.add(Embedding(input_dim = VOCABULARY_SIZE,
# length of vector with which each word is represented
 output_dim = EMBEDDING_SIZE,
# length of input sequence
 input_length = MAX_SEQ_LENGTH,
 weights = [embedding_weights],
 trainable = True ))
rnn_model.add(SimpleRNN(64,return_sequences=True))
# add time distributed (output at each sequence) layer
rnn_model.add(TimeDistributed(tf.keras.layers.Dense(191, activation='softmax')))
#compile model
rnn_model.compile(loss      =  'categorical_crossentropy',
                  optimizer =  'adam',
                  metrics   =  ['acc'])
rnn_model.summary()
```

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_7 (Embedding)     (None, 100, 300)          16954800

 simple_rnn_6 (SimpleRNN)    (None, 100, 64)           23360

 time_distributed_2 (TimeDis  (None, 100, 191)         12415
 tributed)

=================================================================
Total params: 16,990,575
Trainable params: 16,990,575
Non-trainable params: 0
```

3.4 Feature extraction from text

Feature extraction is the process of transforming raw data into a set of numerical features that can be used as input to machine learning algorithms. In text analysis, feature extraction is particularly important because natural language data is inherently unstructured and high-dimensional. Feature extraction is a crucial step in text processing because it allows us to transform unstructured text data into a structured format that can be easily processed and analyzed by machine learning algorithms. Dimensionality reduction is one importance of feature extraction in text processing. Text data is intrinsically multidimensional, which means that each sentence or document may include several words and may have various alternative interpretations. By choosing the most pertinent features that accurately represent the key aspects of the text, feature extraction techniques enable us to minimize the dimensionality of the data. The other factor in feature extraction is improvement in performance. When the input characteristics are clear and informative, machine learning algorithms often perform better. In order to enhance the performance of machine learning models, we may extract useful information from text data, such as the presence of specific words, the frequency of specific patterns, or the similarity of other articles. It is vital to comprehend how the machine learning model arrived at its predictions or classifications in some applications as well. We may build interpretable features with the use of feature extraction, such as topic distributions or sentiment scores, that show how the model is handling the text input.

There are various techniques for extracting features from text data, some of which are given below:

- One Hot Encoding
- Numeric Encoding
- Bag of Words Model
- Bag of N-Grams Model
- TF-IDF Model
- Advanced Word Vectorization Models (Word embeddings)

From the feature extraction techniques given in the above list TF-IDF model and Word embeddings are used in the analysis of Azerbaijani corpus.

The common feature extraction method TF-IDF (Term Frequency-Inverse Document Frequency) is used in natural language processing to convert text input into a format that machine learning algorithms can readily understand. The fundamental goal of TF-IDF is to identify significant words or concepts in a text based on their frequency and the frequency with which they appear in all of the corpus's papers.

The TF-IDF score for a term in a document is produced by multiplying the term frequency (TF) and the inverse document frequency (IDF). Inverse document frequency is how frequently or infrequently a word appears across all documents in a corpus. word frequency is the frequency at which a phrase appears in a document. The IDF factor is calculated as the logarithm of the total number of documents in the corpus divided by the number of documents containing the term.

$$TF - IDF(t, d) = TF(t, d) * IDF(t) \quad (3.1)$$

In this formula

$$TF(t,d) = \frac{number\ of\ times\ t\ appears\ in\ d}{total\ number\ of\ terms\ in\ d} \quad (3.2)$$

Where t is term and d is document. And

$$idf(t) = 1 + log\frac{C}{1 + df(t)} \quad (3.3)$$

Where idf(t) -inverse document frequency for the term t
C - total number of documents in corpus
df(t) - number of documents in which the term t is present

Using TF-IDF for feature extraction in text processing has a number of benefits. First off, it is a quick and efficient approach to vectorize text input so that machine learning algorithms can handle it quickly. Second, it considers both a term's frequency inside a document and its frequency across all papers in a corpus, which aids in the identification of crucial terms that are peculiar to a given document or topic. Thirdly, it may be used for many other text processing tasks, including text categorization, information retrieval, and clustering. However, utilizing TF-IDF for feature extraction has significant drawbacks as well. The fact that it does not account for the semantic connections between words, such as synonyms or related phrases, is one of the key problems. Additionally, it ignores factors like word order and context, which are crucial for some text processing jobs. In circumstances when there are uncommon or infrequent phrases that are crucial to the job, TF-IDF may not perform effectively since they may be given a low TF-IDF score. The other method used for feature extraction that avoids limitations of tf-idf is word embedding techniques.

Word embedding is an approach used in machine learning and natural language processing (NLP) that converts words or phrases into vectors of numerical values. It is a means to turn textual information into real-number vectors that can be fed into machine learning algorithms. The machine can comprehend the meaning of words and their links to one another in a particular context thanks to word embedding. Traditionally, each word in a text document was represented by a vector of zeros, save for a single 1 at the index corresponding to that word in the lexicon. This method of representation is known as one-hot encoding. To capture the semantic linkages between words and to conduct mathematical operations between them, this representation must be high-dimensional and sparse. Word embedding solves this problem by transforming words into low-dimensional dense vectors, often with dimensions between 50 and 300. Word embeddings can be produced in several methods, but they are always the result of modeling a sizable body of text. Using a neural network with an embedding layer, which converts words into vectors, is one common technique. When given a target word or vice versa, the embedding layer is taught to predict the words in the context. The embedding layer's weights are adjusted during training so that the embeddings for words that are similar grow more similar, while the embeddings for words that are different become more dissimilar. After training, we may encode

words as vectors by using the learnt embeddings. The fact that word embeddings are not always comprehensible presents a problem. This means that we can't always interpret an embedding vector's meaning in terms of the word it stands for. The embeddings can still be used to carry out beneficial activities like sentiment analysis or text categorization.

Another difficulty is that word embeddings might not fully capture a word's meaning. For instance, even if homonyms (words with the same spelling but distinct meanings) have different meanings, they may nevertheless have the same embedding. Additionally, the embeddings could be biased when applied to downstream tasks because of the corpus they were trained on. Despite these difficulties, word embeddings are an effective tool for NLP and have been included into several cutting-edge models. They have completely changed the area of NLP by making it feasible for us to complete previously unthinkable tasks like question answering and machine translation. It is expected that word embeddings will continue to be a crucial part of many models as NLP research develops.

A well-liked word embedding approach called Word2Vec is used to convert words into vectors. Since its debut in 2013 by Tomas Mikolov and his Google team, it has grown to rank among the most popular models for natural language processing. In order for Word2Vec to predict the context of a given word or vice versa, a neural network must first be trained. The model parses a vast corpus of literature and builds a vocabulary from each word. Then, each word is represented as a vector in a high-dimensional space, with the dimensions denoting the word's many attributes. Word2Vec uses the Continuous Bag of Words (CBOW) model and the Skip-Gram model as its two primary architectures. While the Skip-Gram model predicts the context words based on the target word, the CBOW model predicts the target word based on the context words. (Figure 5) A basic neural network with one hidden layer serves as the foundation for both topologies.

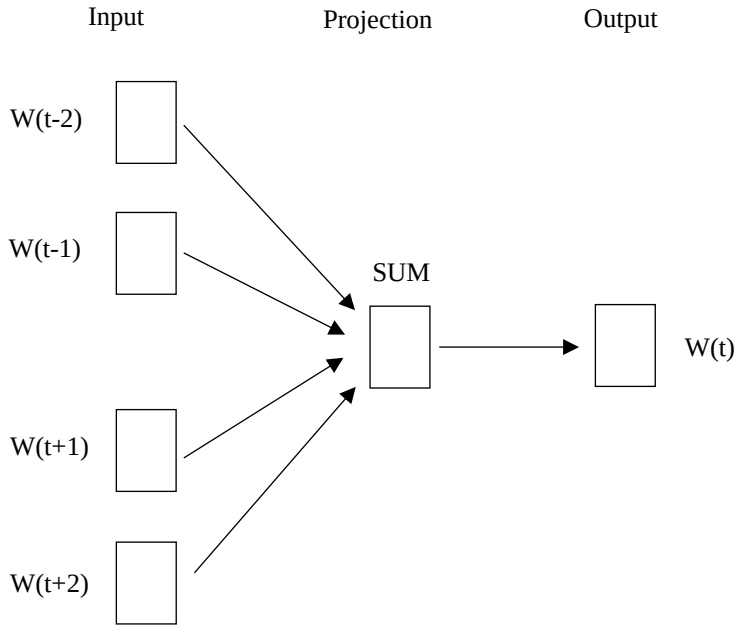

Figure 6: CBOW model of Word2Vec

Backpropagation is a method for updating the neural network's weights during training that reduces the discrepancy between the expected and actual output. The actual output is shown as a one-hot vector, with all other locations set to 0, and just the position corresponding to the right

word set to 1. Each location of the projected output's vector of probabilities corresponds to a word from the dictionary. The hidden layer's weights are employed as the word embeddings after training. Natural language processing activities like sentiment analysis, text categorization, and machine translation may all be performed using these embeddings. The capacity of Word2Vec to identify semantic links between words is one of its main benefits. In the embedding space, the vectors for words with similar meanings are grouped together, while those for words with different meanings are spread out. In addition, the embeddings provide vector arithmetic, allowing one to combine or subtract one word's vector from another word's vector to produce a new vector that illustrates the semantic link between the two words. For instance, the vector for "king" closest to the vector for "queen" is the vector for "king" minus the vector for "man" plus the vector for "woman". Because of its efficiency and scalability, Word2Vec has gained popularity as a tool for natural language processing. Large text datasets may be utilized to train the model, and the resulting embeddings can be used to a variety of applications. Word2Vec is also computationally effective, making it useful for usage in actual applications. I used genism library for implementing Word2Vec word embedding in python (Code 1). These embeddings are used in two functionality of corpus analysis. One is word similarity that converts words to vector space with the predefined dimension then calculates cosine similarity between vectors the other use case is POS tagging. In the Recurrent Neural Network POS tagging architecture inputs to the model are converted to numeric format by Word2Vec embedding then results pushed to the model. Applying word embedding before training model increased both training and validation accuracy significantly.

Code 6: Word2Vec embedding for word similarity

```
from gensim.models import Word2Vec
import pickle
model = Word2Vec(tokens_for_vec,vector_size=100, window=5, min_count=1, workers=4)
filename = 'Word2Vec_model.sav'
pickle.dump(model, open(filename, 'wb'))
```

```
1 model.wv.most_similar("azərbaycan")
```

```
[('türkiyə', 0.6591471433639526),
 ('qırğız', 0.6510428786277771),
 ('koreya', 0.6458494663238525),
 ('belarus', 0.6331899166107178),
 ('azərbaycanın', 0.6289570927619934),
 ('çex', 0.6165573000907898),
 ('dağıstan', 0.6028293967247009),
 ('tatarıstan', 0.594680905342102),
 ('polşa', 0.5920835733413696),
```

Dimension of vectors are defined as 100.

Another word embedding technique used in this paper is FastText model. FastText is a word embedding generation method based on neural networks that was created by Facebook's AI Research team. FastText expands on Word2Vec, except it produces subword embeddings rather than embeddings at the word level. This implies that each word is represented as a group of character n-grams, and the word's embedding is created by combining the embeddings of these subwords. FastText's ability to capture word morphology over Word2Vec is one of its primary benefits. FastText can learn representations for uncommon or seldom used words as well as words with different prefixes and suffixes by modeling the subwords of a word. This is particularly helpful for morphologically complicated languages where a single word may have several inflections and conjugations. On a number of natural language processing tasks, including sentiment analysis, text classification, and named entity identification, FastText has been found to perform better than Word2Vec. FastText embeddings have also been utilized in machine translation, where it has been demonstrated that they enhance the quality of translation for languages with limited resources. In my research, where I used word embedding techniques for word similarity measure, Fast-Text and Word2Vec gave almost the same results, but FastText embeddings provide a better representation of rare or unseen words in our dataset, which could lead to more accurate results in the analyses. The greater complexity of modeling subwords in FastText, however, needs more processing effort and training data than Word2Vec. So, it's crucial to carefully weigh the trade-offs between utilizing Word2Vec and FastText.

3.5 N-grams

N-grams are a collection of N consecutive words that are taken from a corpus of text in computational linguistics. Ngrams are a core idea in natural language processing and have several uses in information retrieval, machine translation, and language modeling. N in Ngrams can range from 1 to infinite, with the most frequent values being unigrams, bigrams, trigrams, and 4-grams (N=1, 2, 3, and 4 respectively). A corpus's frequency and distribution of words and phrases may be determined using ngrams, which offer a statistical representation of language usage. Ngrams, for example, can be used to create language models that forecast the likelihood of a sequence of words given the context or to compute the frequency of a particular word or phrase in a corpus.

Recent years have seen a rise in the usage of Ngrams in natural language processing, notably with the introduction of deep learning-based techniques like Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). These models may be utilized for many NLP applications including sentiment analysis, text classification, and machine translation by using ngrams to produce the input characteristics.

Higher-order Ngrams, however, can produce a lot of features, which can cause overfitting and other computational issues. Therefore, it is crucial to strike a balance between the advantages of employing higher-order Ngrams and the computational and statistical difficulties they provide.

In this research Ngrams range from 1 (unigrams) to 6 (six-grams). Some examples are given in Table 2. Again we used python nltk library for ngram analysis (Code 1).

Code 7: Bigrams

```
import pickle
from nltk.util import ngrams
from collections import Counter
#loading tokens
with open('clean_tokens.pkl', 'rb') as f:
    clean_tokens = pickle.load(f)
#removing stop words
clean_tokens=[i for i in clean_tokens if i not in stop_words]

bigrams = ngrams(clean_tokens, 2)
bigram_freq = Counter(bigrams)
#sorting according to the frequency
sorted_counter = sorted(bigram_freq.items(), key=lambda x: x[1], reverse=True)

```

Table 2: Examples to Bigrams

Part1	Part2	Frequency
azərbaycan	respublikası	95532
kənd	təsərrüfatı	30162
bakı	şəhəri	26427
icra	hakimiyyəti	23103
həsr	olunmuş	14943
fəaliyyət	göstərən	19418
ümummilli	lider	13661
havanın	temperaturunun	13666
əməliyyataxtarış	tədbirləri	13397
təbii	sərvətlər	13357

4. RESEARCH RESULTS AND ANALYSIS OF RESULTS

After comprehensive research and implementations, we have created a sizable Azerbaijani language corpus with a wide variety of functions following extensive study and analysis. Results of analysis are delivered for implementation of friendly User Interface for enabling interaction of users. It offers the following characteristics, which allow for a thorough language analysis: Word Frequency, Ngrams, Concordance, Thesaurus, and Word Sketch.

One of the main functionalities of the project was POS tagging model. Our HMM model resulted in 91.9% training accuracy. Then model is tested with test dataset that was created by splitting

main dataset to training and testing part in 80%-20% ratio. In test dataset model preformed very low accuracy score with only 45.15%. Such huge difference between test and train dataset indicates that model failed to generalize the complex pattern in the datasets and caused overfitting in training section.

Our RNN model which was described in the previous section was also trained with the same POS tagged training data. And accuracy results were 98% and 97% for training and validation respectively (Figure 6). Selection of this ratio was due to the large available dataset as even 20 percent of it was considered as reliable for testing.

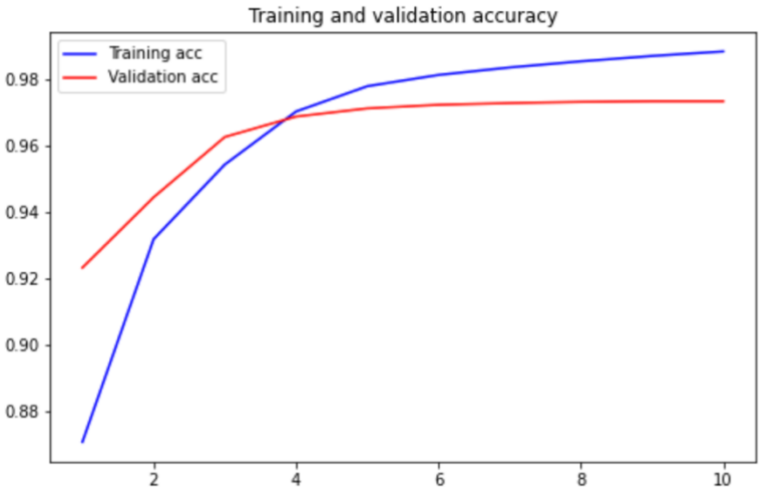

Figure 7: RNN POS tagging model train and validation accuracy

Another functionality of corpus was visualization of lexical dispersion plot. A lexical dispersion plot is a visual tool used in corpus linguistics to examine how a certain word or group of words are distributed throughout a corpus of text. The graphic shows how words appear over time, exposing use and co-occurrence patterns in a text or collection of texts. The generated figure may be used to see how frequently the target words appear together and in what patterns. A jagged line on the plot, on the other hand, denotes that the term occurs in clusters or is used more frequently in particular portions of the text, while a flat line on the plot indicates that the word appears uniformly across the text. Such kind of analysis is especially important on spoken language to detect which words appear in the beginning of the speech and which occurs mostly at the end of speech. The Given example below (Figure 7) represents how words ‘hüquq’,’kitab’,’insan’,’nümunə’ are distributed across the corpus.

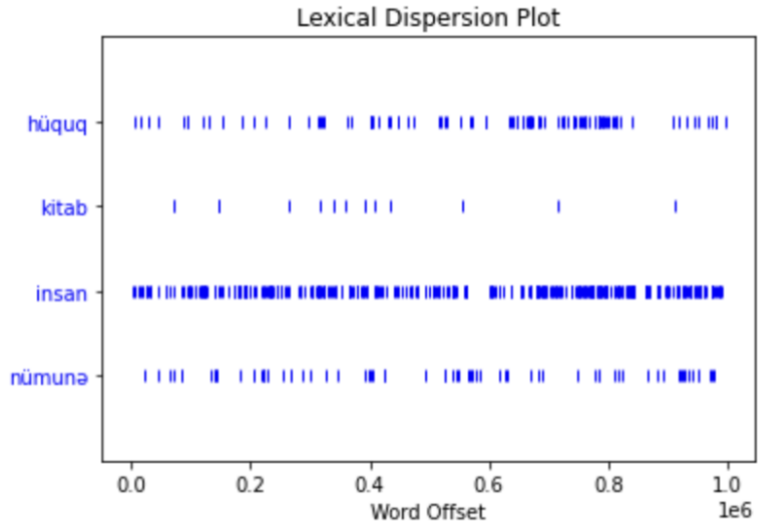

Figure 8: Lexical Dispersion plot

Another functionality was concordance analysis. In corpus linguistics, the technique of concordance analysis is employed to examine how a certain term or phrase is used throughout a corpus of text. During the analysis, a concordance—a list of every instance of a target word or phrase in a text, together with the words that come before and after it in context—is created. A word or phrase's frequency of use, its collocations (words that frequently appear with it), and its semantic connections (words that frequently occur in related contexts) may all be learned using concordance analysis. Additionally, the study might shed light on a text or corpus's stylistic and rhetorical characteristics. The study of language learning and usage is one area where concordance analysis is put to use. Researchers can determine the usual settings in which a target word is used, the frequency and distribution of its usage, and the collocations that are most closely related with it by looking at the concordance of the word. The creation of more effective vocabulary education and learning materials may then be done using this knowledge. Concordance analysis is also used to examine how languages develop and evolve. Researchers can spot changes in a word or phrase's frequency, distribution, and collocations as well as gain understanding into the mechanisms behind these changes by examining the concordance of that term or phrase over several historical periods or geographies. Below given the screen from our corpus UI with the concordance of the word ‘Azərbaycan’ (Figure 8).

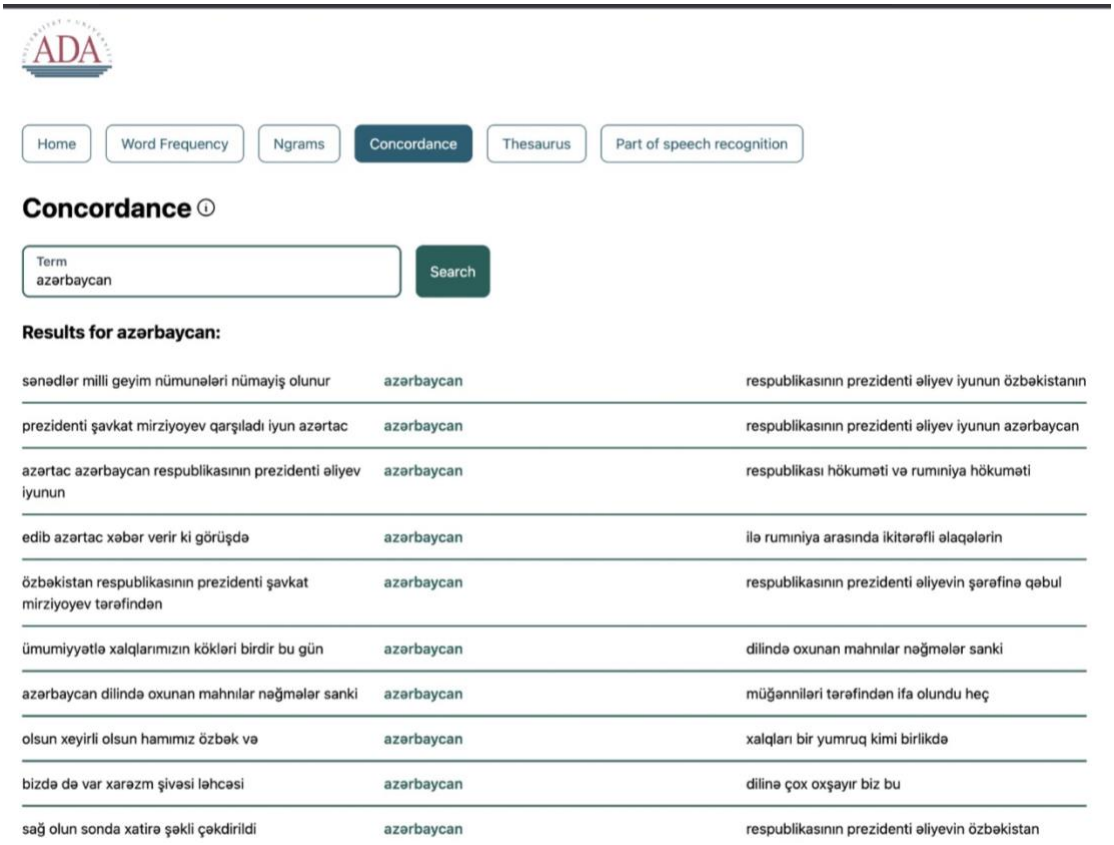

Figure 9: Concordance analysis page from UI of the corpus

5. SUMMARY AND FUTURE WORK

In conclusion, the goal of this thesis was to give a thorough study of the Azerbaijani language corpus that included both quantitative and semantic analysis as well as the development of more precise POS tagging models utilizing Hidden Markov Model and Long Short-Term Memory. The constructed models, with the LSTM model obtaining a 97% accuracy rate on test data, considerably increased the accuracy of POS tagging in the Azerbaijani language, according to the research. The study also emphasized the study's constraints and presumptions, such as the possibility of linguistic evolution through time and the requirement for more varied and inclusive data.

There are various potential areas for further investigation. First, the corpus analysis may be expanded to incorporate a wider range of texts, such as social media data, which can shed light on the development and usage of language today. To further boost the accuracy of these tasks in Azerbaijani language, the work may also be applied to other NLP tasks like named entity identification or sentiment analysis. The study can also investigate the efficacy of various deep

learning models for POS tagging and other NLP tasks in Azerbaijani language, such as Convolutional Neural Networks or Transformers. The study can also look at how unsupervised learning methods, including clustering and topic modeling, might be used to explore the semantic structure of Azerbaijani language and spot trends in the data.
As a future work, Azerbaijani corpus would be combined from different areas and encompasses various use cases of the language. It should be noted that not only written samples of language should be used in the corpus development, spoken samples also should exists in the corpus. General categorization of corpus could be like that: main 6 categories are -spoken language, fiction, literature, news, academic and web. Each of the main categories should be represented with equal percentage of total words in the corpus (Table 2).

Table 2: Categories and the ratios of the corpus

Category	***Ratio***
Spoken	16%
Fiction	16%
Literature	16%
News	16%
Academic	16%
Web	16%

Finally, the study can aid in the creation of more sophisticated natural language processing resources and tools for the Azerbaijani language, which will be advantageous to many sectors of the economy, including the media, education, and communication. The availability of such materials can help encourage the use of the Azerbaijani language and its preservation, particularly in light of the language's rising use in digital media and communication platforms. Overall, this thesis offers important contributions to the study of Azerbaijani language processing and opens up fresh directions for future study.

6 REFERENCES

[1] ‘How to use statistics in quantitative corpus analysis’ Stefan Th. Gries University of California, Santa Barbara & Justus Liebig University Giessen, 2019
[2] ‘A Zipfian Approach to Words in Contexts: The Cases of Modern English and Chinese’, Jin Cong, 2021
[3] ‘Contextual correlates of semantic similarity, Language and Cognitive Processes’George A. Miller & Walter G. Charles, 2013
[4] ‘Frequency Estimates for Statistical Word Similarity Measures’, Egidio Terra, C. L. A. Clarke, 2013
[5] ‘Second Order Co-occurrence PMI for Determining the Semantic Similarity of Words’, Md. Aminul Islam and Diana Inkpen, 2015
[6] ‘Second-order Co-occurrence Sensitivity of Skip-Gram with Negative Sampling’, Dominik Schlechtweg, Cennet Oguz, Sabine Schulte im Walde, 2014
[7] ‘Zipf’s word frequency law in natural language: A critical review and future directions’, Steven T. Piantadosi, October 2014
[8] ‘The Statistics of Text: New methods for Content Analysis’, Will Lowe, 2016
[9] ‘The 385+ million word Corpus of Contemporary American English’, Mark Davies, 2013
[10] ‘The Corpus of Contemporary American English as the first reliable monitor corpus of English’, Mark Davies, 2013
[11] ‘Comparing Lexical Bundles across Corpora of Different Sizes: The Zipfian Problem’, Yves Bestgen, February 2019
[12] ‘Leipzig Corpus Miner - A Text Mining Infrastructure for Qualitative Data Analysis’, Andreas Niekler, Gregor Wiedemann, Gerhard Heyer, June 2014
[13] ‘Large Corpora for Turkic Languages and Unsupervised Morphological Analysis’ , V´ıt Baisa, V´ıt Suchomel, 2014
[14] ‘On the Relative Influence of Corpus and Dictionary Size in a Study Using Non-Parallel Corpora’, Oliver Cromm, August 2013
[15] “AzWaC – Azerbaijani Corpus from the Web.” Sketch Engine, 20 Feb. 2018, https://www.sketchengine.eu/azwacazerbaijani-corpus/
[16] Davies, M. 2009. “Word frequency in context: Alternative architectures for examining related words, register variation and historical change”. In D. Archer (Ed.), What’s in a Word-list? Investigating Word Frequency and Keyword Extraction. London: Ashgate, 53–68.
[17] “NLTK :: Natural Language Toolkit”, [Online]. Available: https://www.nltk.org. [Accessed: 6-Apr.-2023].
[18] “Sketch Engine”, [Online]. Available: https://app.sketchengine.eu/. [Accessed: 12-Apr.-2023].
[19] ‘“English-Corpora: COCA”, [Online]. Available: https://www.english-corpora.org/coca/. [Accessed: 12-Apr.-2023].
[20] “Word Frequencies in Written and Spoken English: based on the British National Corpus.” (Geoffrey Leech, Paul Rayson, Andrew Wilson), ISBN 0582-32007-0