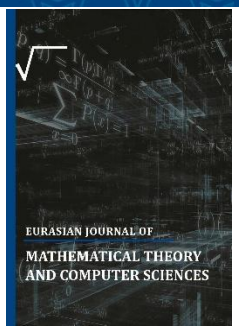
EURASIAN JOURNAL OF MATHEMATICAL THEORY AND COMPUTER SCIENCES
Innovative Academy Research Support Center
IF = 9.206 www.in-academy.uz/index.php/ejmtcs



THREE-STAGE APPROACH TO TEXT DATA PROCESSING AND ITS ALGORITHMS

Rakhmanov Askar
Department of "System and Applied Programming" Tashkent University of Information Technologies named after Muhammad al-Khwarizmi. asqartr1.2.3dipu@gmail.com
Abduvalieva Zebiniso
Department of "System and Applied Programming" Tashkent University of Information Technologies named after Muhammad al-Khwarizmi zebinosoabduvaliyeva@gmail.com
https://doi.org/10.5281/zenodo.17909167

ARTICLE INFO
Received: 05th December 2025
Accepted: 11th December 2025
Online: 12th December 2025
KEYWORDS
Lemmatization, tokenization, stemming, text processing, text normalization, machine learning, text classification, language processing algorithms, NLP, text classification by category.

ABSTRACT
This article examines the main methods of text data processing: lemmatization, tokenization, and stemming. These methods are used to normalize and prepare text for analysis and machine learning. The algorithms and approaches for implementing each method are described, and their advantages and disadvantages are analyzed. The research results guide the selection of an appropriate method based on the task and the characteristics of the text being processed.

ТРЁХЭТАПНЫЙ ПОДХОД ОБРАБОТКИ ТЕКСТОВЫХ ДАННЫХ И ЕГО АЛГОРИТМЫ

Рахманов Аскар
Ташкентский университет Информационных технологий
Кафедра “Системное и прикладное программирование”
asqartr1.2.3dipu@gmail.com
Абдувалиева Зебинисо
Ташкентский университет Информационных технологий
Кафедра “Системное и прикладное программирование”
zebinosoabduvaliyeva@gmail.com
https://doi.org/10.5281/zenodo.17909167

ARTICLE INFO
Received: 05th December 2025
Accepted: 11th December 2025
Online: 12th December 2025

ABSTRACT
В данной статье рассматриваются основные методы обработки текстовых данных: лемматизация, токенизация и стемминг. Эти методы используются для нормализации и подготовки текста к анализу и машинному обучению. Описаны алгоритмы и подходы к реализации каждого метода, проанализированы их преимущества и недостатки. Результаты исследований приводят к выбору подходящего

EURASIAN JOURNAL OF MATHEMATICAL THEORY AND COMPUTER SCIENCES
Innovative Academy Research Support Center
IF = 9.206 www.in-academy.uz/index.php/ejmtcs

KEYWORDS

Лемматизация, токенизация, стемминг, обработка текста, нормализация текста, машинное обучение, классификация текста, алгоритмы обработки языка, НЛП, классификации текстов по категориям.

метода в зависимости от задачи и характеристик обрабатываемого текста.

Обработка текстовых данных состоит из трех последовательных этапов: устранение ошибок, лексическая нормализация слова и сведение к основе. Устранение грамматических и лексических ошибок - это сложная процедура, для выполнения которой нужно провести большое количество исследований [1]. Следует учитывать не только правила нужного языка, но и множество исключений. Один из самых сложных задач, связанных с обработкой естественного языка, это понимание семантики слов, точнее, формирование признаков таким образом, чтобы алгоритм имел возможность различать понятия, а не наборы букв. Во время векторизации текст превращается в наборы числовых признаков, которые отражают его особенности: синтаксис, семантику, значение. Модели машинного обучения используют эти числовые признаки, чтобы решать такие задачи как анализ структуры предложений и разбиение на составные части, классификация текстов, извлечение информации, выделение ключевых данных и смыслов из неструктурированных текстов.

Машинный перевод с учётом собственного языка означает, что модель не просто заменяет слова из одного языка словами из другого, а учитывает грамматическую структуру, порядок слов, идиомы, фразовые выражения и контекст, чтобы сохранить смысл и стиль исходного текста. Генерация текстовых данных — это процесс создания связных и осмысленных текстовых последовательностей с помощью компьютерных моделей, таких как нейросети и языковые модели. Эти модели анализируют тексты большего объема и учатся предсказывать следующее слово или фразу, благодаря чему есть возможность создавать новые тексты, похожие по стилю и смыслу на обучающие данные. Извлечение информации-модель получает на вход неструктурированный текст и выделяет из него ключевые слова, термины и основные идеи текста. При решении любой из этих задач модель использует числа, а не слова в чистом виде. Поэтому векторизация важна для каждого направления NLP, без нее невозможно представить обработку текста естественного языка на компьютере[1-5]. При классификации текстовых данных и ключевых слов задачи NLP включают в себя классификации текстов по категориям: определение тематики документа, жанра, замысла автора, тональности (позитивная, негативная, нейтральная). Выделение ключевых слов и фраз, автоматический поиск важных терминов и понятий,

которые характеризуют содержание текста. Определение смысловых меток, автоматическая маркировка текстов для фильтрации и быстрого поиска. Классификация по настроению (анализ настроений), проявление эмоциональной окраски текста. Аннотация и категоризация побочных обзоров, сообщений в соцсетях, новостей, научных статей. Выделение сущностей и их классификация, для создания метаданных. Построение тематических моделей для сохранения скрытых тем в больших корпусах текстов.

Формирование признаков в задачах обработки текстовых данных играет ключевую роль, так как она определяет, какие характеристики текста будут использоваться для анализа и построения моделей. Это этап преобразования текста в числовое представление, относящееся к машинному обучению. Одним из подходов к созданию признаков является выделение различных характеристик текста [3]. К признакам текстовых данных относится частота слов в предложение или документе. Под частотой следует понимать общее количество слов в тексте. Длиной текста в предложение или в документе следует считать число символов или слов. Кроме того, при формировании признаков текстовых данных учитывается средняя длина слова, которая рассчитывает среднее арифметическое всех слов в тексте. Количество уникальных слов в предложении, это оценка разнообразия текста. При формировании признаков текста следует учитывать исключение так называемых стоп слов, к которым относятся слова, не имеющие особого значения. Для генерации числовых и текстовых признаков из текста применяются такие методы как TF-IDF, N-граммы, векторные представления слов, с помощью моделей Word2Vec, GloVe, FastText, расчет вероятностей, происходящих последовательно слов обработки текста, которые преобразуют текстовые данные в числовую форму для последующего анализа или применения в моделях машинного обучения [4].

К числовым признакам можно отнести частотность слов - количество вхождений каждого слова в текст или корпус, взвешенная частота слов, наблюдаемая его распространенность в документах, периодически последовательность из N слов для учета контекста, Длина текста - количество слов, символы, предложения, длина слов- средняя длина слов в тексте. К текстовым признакам можно отнести леммы и части речи, ключевые слова и фразы, удаление стоп-слов, слов, которые не влияют на смысл предложения и не имеют большую значимость. Обычно система, выполняющая замену слов и удаление стоп-слов, смотрит смысл. Отсюда следует, что стоп-слова, влияющие на смысл, особенно при обратных заменах, не удаляются и обрабатываются аккуратно, чтобы не исказить смысл. При замене слов «в прямом смысле», слово заменяется на синоним или другое слово из словарной замены, которое сохраняет смысл или близко к исходному. Замена слов «в обратном смысле», обращение значения, например, отрицание, антоним играет важную роль для сохранения или изменения смысла предложения. В этом случае удаление стоп-слов помогает не потерять важные смысловые части, например, слова-отрицания «не», «нет», которые не должны быть удалены, так как изменяют смысловые выражения [5].

Нормализация слов - это задача приведения слов или токенов к стандартному формату. Токен это отдельное слово или символ. Токенизация, процесс разбиения текста на отдельные элементы (токены). Слова, токены, знаки препинания, числа. Пример текста: «Я люблю программировать на Python!». Токены:[“Я”, “люблю”, “программировать”, “на”, “Python”, “!”»]. Алгоритм токенизации состоит из следующих этапов. На первом этапе на вход подаются необработанные текст. На втором этапе происходит очистка текста от лишних пробелов, нормализация кавычки, дефисов, символов. На следующем этапе разделяют текст используя пробелы, знаки препинания, табуляцию и др. После разделения наступает этап фильтрации токенов, исключение пустых строк, лишних символов.

Самым простым случаем нормализации является приведение к нижнему регистру. Приведение всех символов к нижнему регистру означает, что слова *Ta’lim* и *talim* будут представляться одинаково, что является весьма выгодным для обобщений во многих задачах, таких как извлечение информации или распознавание речи. Для анализа задач классификации текста, извлечения информации и машинного перевода, наоборот, регистр является полезным, и обычно применение к нижнему регистру не применяется [6]. Иногда мы создаем как версии с сохранением регистра (то есть с учетом как заглавных, так и строчных букв), так и без регистра для языковых моделей. Также не приводят к регистру, если важен регистр для понимания. В некоторых странах регистратор влияет на грамматическую роль или ударение. Приводит к верхнему регистру, если необходимо выделить или стандартизировать текст, например, при оформлении заголовков, табличных данных. Верхний регистр часто используется для визуального акцента, например, при написание собственных имен, сокращений, кодов или авторских прав.

Нормализация текста или лемматизация означает преобразование его в более удобную, стандартную форму. Например, большая часть того, что мы собираемся делать с языком, основана на первом шаге - выделении или токенизации слов или частей слов из беглого текста, задаче токенизации.

Системы, использующие BPE (Byte Pair Encoding), метод сегментации текста, который итеративно находит самые частные пары символов (байтов) в тексте и выбирает их в новый токен (подслово) или другие методы токенизации снизу вверх, могут не выполнять дальнейшую нормализацию слов. Здесь речь идёт о системах токенизации текста, которые используют BPE (Byte Pair Encoding) — это метод, который шаг за шагом (итеративно) находит наиболее часто встречающиеся пары символов в тексте и объединяет их в один новый токен (подслово). Такие системы обычно не проводят дополнительную нормализацию слов (например, не приводят слова к нижнему регистру, не удаляют окончания и т. п.), потому что сама процедура BPE уже определяет структуру подслов. Процесс повторяется до достижения нужного размера словаря или других критериев. Это разбиение слов на более важный фактор, который полезен для обработки переменных и неизвестных слов, позволяет повысить эффективность и гибкость языковых моделей.

В других системах обработки естественного языка может потребоваться дополнительная нормализация, например, выбор одной нормальной формы для слов с несколькими формами, таких как *UZ* и *uz* или *uh-huh* и *uhhuh*. Такая стандартизация может быть полезной, несмотря на потерю информации о правописании в процессе нормализации. В обработке естественного языка токенизация - это особый вид сегментирования документов. При сегментации текст разбивается на мелкие куски (сегменты) с более узким информационным содержанием [1-7].

Сегментация может включать разбиение документа на абзацы, те на предложения, их на фразы и последние — на токены (слова), а также знаки препинания. Первым этапом в процессе обработки естественного языка является токенизация и она может серьезно повлиять на остальные этапы. Модуль разбиения на токены обрабатывает неструктурированные данные — текст на естественном языке, разбивая их на фрагменты информации, которые можно считать отдельными элементами. Такие подсчитанные количества вхождений токенов в документ можно непосредственно использовать как представляющий этот документ вектор. Подобный подход позволяет сразу получить из несформированной строки текстового документа, числовую структуру данных, подходящую для машинного обучения. Их можно использовать в многошаговом процессе построения моделей машинного обучения в качестве критериев, запускающих систему для обработки слов, более сложных решений или поведений.

Сегментация предложений — это еще один основной этап в обработке текста. Наиболее полезными признаками для сегментации текста на предложения являются знаки препинания, такие как точки, вопросительные знаки и восклицательные знаки. Вопросительные знаки и восклицательные знаки относительно однозначно указывают на границы предложений. Точка, с другой стороны, более неоднозначна. Символ точки "." может быть как маркером границы предложения, так и маркером сокращений, таких как *Mr.* или *Inc.*. Предыдущее предложение, которое вы только что прочитали, показало еще более сложный случай этой неоднозначности, когда последняя точка в *Inc.* отмечала как сокращение, так и границу предложения. По этой причине токенизация предложений и токенизация слов могут быть решены совместно[5].

Для других задач в обработке естественного языка также требуется, чтобы две разные формы слова вели себя одинаково. Например, при веб-поиске кто-то может ввести строку *adabiyotlar*, но полезная система захочет также вернуть страницы, которые упоминают слова *adabiyot*, сведение слова к его базовой форме - это задача определения того, что два слова имеют общий корень, несмотря на их поверхностные различия. Например, слова *nonushta* и *non* оба имеют лемму *non*. Лемматизация этих форм в одну и ту же лемму позволяет нам найти все упоминания таких слов, сведение слова к его базовой форме предложения, как например «U kitobni uqiyapti», будет выглядеть как «U kitob uqish». Леммитизация процесс приведения слов к его месту или начальной форме (лемме).

Наиболее продвинутые методы сведения слова к его базовой форме предполагают проведение полного разбора формы и грамматических характеристик слова, анализа слова, то есть процесс изучения и составления слов на его минимально значимую единицу текста с определением грамматических и лексических характеристик слов. Морфология занимается изучением структуры слов, которые состоят из более мелких значимых элементов - морфем. Составные компоненты слова делятся на две основные категории: основа, являющаяся главной частью слова и несущая его основное значение, и аффиксы, которые добавляют дополнительные смысловые оттенки [6]. К примеру, слово ta'lim содержит только одну морфему, а в слове ta'limning уже две морфемы: ta'lim и ning. Морфологический анализатор разбивает слово на составляющие ta'lim и ning.

Алгоритмы сведения слова к его базовой форме могут быть сложными. По этой причине иногда мы используем более простой, но грубый метод, который в основном заключается в отрезании окончаний слов. Эта упрощенная версия разбора формы и грамматических характеристик слова называется стемминг. Также стемминг можно рассматривать как способ нахождения основ слов, способа удаления приставок, суффиксов и окончаний. Например, классический стеммер Портера суть которого заключается в последовательном наборе правил для обрезки суффиксов и окончаний в словах с целью получения их основ, при применении к следующему абзацу: davlat hokimiyati tizimida Oliy Majlisning rolini oshirish, uning mamlakat ichki va tashqi siyosatiga oid muhim vazifalarni hal etish hamda ijro hokimiyati faoliyati ustidan parlament nazoratini amalga oshirish bo'yicha vakolatlarini yanada kengaytirish, дает следующий стеммированный результат: davlat hokimiyat tizim Oliy Majlis rol oshirish, u mamlakat ichki va tashqi siyosati oid muhim vazifa hal etish hamda ijro hokimiyati faoliyati usti parlament nazorati amalga oshirish bo'yicha vakolat yana kengaytirish [2].

Алгоритм основывается на правилах перезаписи, которые выполняются последовательно, при этом результат каждого прохода подается на вход следующего, hokimiyati- hokimiyat, vazifalarni → vazifa. Простые стеммеры оказываются полезными, когда необходимо объединить различные формы одного и того же слова. Однако в современных системах их применяют реже, поскольку они склонны допускать ошибки как из-за чрезмерного обобщения, так и из-за недостаточного охвата.

Методы токенизации предложений в первую очередь решают на основе правил или машинного обучения, является ли точка частью слова или маркером границы предложения. Словарь сокращений может помочь определить, является ли точка частью часто используемого сокращения; такие словари могут быть созданы вручную или с помощью машинного обучения, как и финальное разбиение предложений.

Минимальное расстояние редактирования - это математически строго определенная мера для измерения различия между строками на основе операций редактирования, определяющая, насколько одна строка отличается от другой [6]. Минимальное расстояние редактирования, метрика сходства между двумя

строковыми последовательностями, еще один способ редактирование и приведение к нужной форме. Чем больше расстояние, тем более различны строки. Расстояние редактирования дает способ количественно оценить сходство строк. Формально минимальное расстояние редактирования между двумя строками определяется как минимальное количество операций редактирования (операций, таких как вставка, удаление, замена), необходимых для преобразования одной строки в другую. Дано два последовательных символа, согласование — это соответствие между подстроками этих двух последовательностей. Таким образом, можно выяснить, что I соответствует пустой строке, N - с E, и так далее. Под согласованными строками представлена еще одна визуализация — серия символов, представляющих список операций для преобразования верхней строки в нижнюю: d для удаления, s для замены, i для вставки. Также можно назначить каждому из этих действий определенную стоимость или вес. стоимость, предполагается, что замена буквы на саму себя, например, t на t, не имеет стоимости.

Расстояние Левенштейна необходимо для измерения разницы между двумя словами или предложениями. Оно показывает, сколько минимальных операций по вставке, удалению или замене символов необходимо выполнить, чтобы превратить одно слово или предложение в другое. Это понятие широко применяется в разных областях. Например, исправление ошибок в тексте (автокоррекция, проверка орфографии), где помогает находить и исправлять опечатки и ошибки ввода. Поисковые системы используют его для поиска похожих слов или фраз, даже если пользователь ошибся при вводе. При обработке естественного языка и машинного перевода, оно используется для сравнения вариантов перевода или для обнаружения степени совпадения или различия текстов. Расстояние Левенштейна помогает понять окончательно, насколько схожи две строки или же какие минимальные изменения их разделяют[6]. Расстояние Левенштейна это минимальное количество операций вставки, снятия и замены, необходимое для преобразования одной строки S_1 в другую строку S_2.

Если обозначить D(i,j) как минимальное количество операций для преобразования первых i символов строки S_1в первые j символов строки S_2, при этом, разрешённые операции: вставка символов (1 шаг), удаление символов (1 шаг), замена символа на другой (1 шаг).

Формальное определение выглядит так: если $S_1[i] = S_2[j]$, тогда $D(i,j) = D(i-1,j-1)$, в противном случае

$$D(i,j) = min \begin{cases} D(i-1,j)+1, & \text{удаление} \\ D(i,j-1)+1, & \text{вставка} \\ D(i-1,j-1)+P(S_1[i],S_2[j]), & \text{замена} \end{cases} \quad (1.1)$$

Под P (1.1) понимается шаг операции замены символа или отсутствие этой замены, то есть если символы на позициях i и j строк S_1и S_2 , совпадают, нет необходимости менять, то $(P = 0)$ и это означает, что замена не требуется. Если символы разные, то шаг замены равна 1 $(P = 1)$ — это значит, что нужно провести

EURASIAN JOURNAL OF MATHEMATICAL THEORY AND COMPUTER SCIENCES
Innovative Academy Research Support Center
IF = 9.206 www.in-academy.uz/index.php/ejmtcs

замену символов. Замена не производится если текущие символи строки не соответствует символу в той же позиции строки, например если пятая буква в строке S_1 не соответствует пятой букве в строке S_2.

Если одна из строк пустая ($i = 0$ или $j = 0$), расстояние равно длине другой строки. Иначе расстояние будет рассчитываться по трем вышеперечисленным формулам. Удаление символа это $D(i-1, j) + 1$, вставка символа это $D(i, j-1) + 1$, замена символа это $D(i-1, j-1) + P(S_1\lceil i\rceil, S_2\lceil j\rceil)$.

Пример вычисления расстояния Левенштейна для строк "mamlakatimizda" и "mustaqillik" с подробным описанием шагов.

Исходные данные строка 1: "mamlakatimizda" (длина 13), строка 2: "mustaqillik" (длина 11).

Шаг 1. Инициализация матрицы. Создается матрица размером (14 x 12), где строки соответствуют символам строки 1 плюс пустому символу в начале, а столбцы - символам строки 2 плюс пустому символу.

Первая строка и первый столбец заполняются последовательными числами от 0: первое значение - 0, дальше счетчик длины.

Шаг 2. Заполнение матрицы. Для каждой ячейки $D(i, j)$ где $i > 0$ и $j > 0$ вычисляем по формуле (1.1)

Шаг 3. Пример нескольких вычислений D: сравниваем 'm' и 'm'

Стоимость замены = 0 (символы совпадают)

$$min\begin{cases} D + 1 = 1 + 1 = 2 \\ D + 1 = 1 + 1 = 2 = 0 \\ D + 0 = 0 + 0 = 0 \end{cases} \quad (1.2)$$

D: ‘m’ и ‘u’

Стоимость замены =1

$$min\begin{cases} D + 1 = 2 + 1 = 3 \\ D + 1 = 0 + 1 = 1 = 1 \\ D + 1 = 1 + 1 = 2 \end{cases} \quad (1.3)$$

D: ‘a’ и ‘u’

$$min\begin{cases} D + 1 = 1 + 1 = 2 \\ D + 1 = 2 + 1 = 3 = 1 \\ D + 1 = 0 + 1 = 1 \end{cases} \quad (1.4)$$

Шаг 4. Заполнение всей матрицы и вывод результата.

По аналогии вычисляются все ячейки матрицы. Ниже приведен только последний элемент матрицы показывающая итоговый результат

Последняя строка/столбец	m	u	s	t	a	q	i	l	l	i	k
a (14)											7

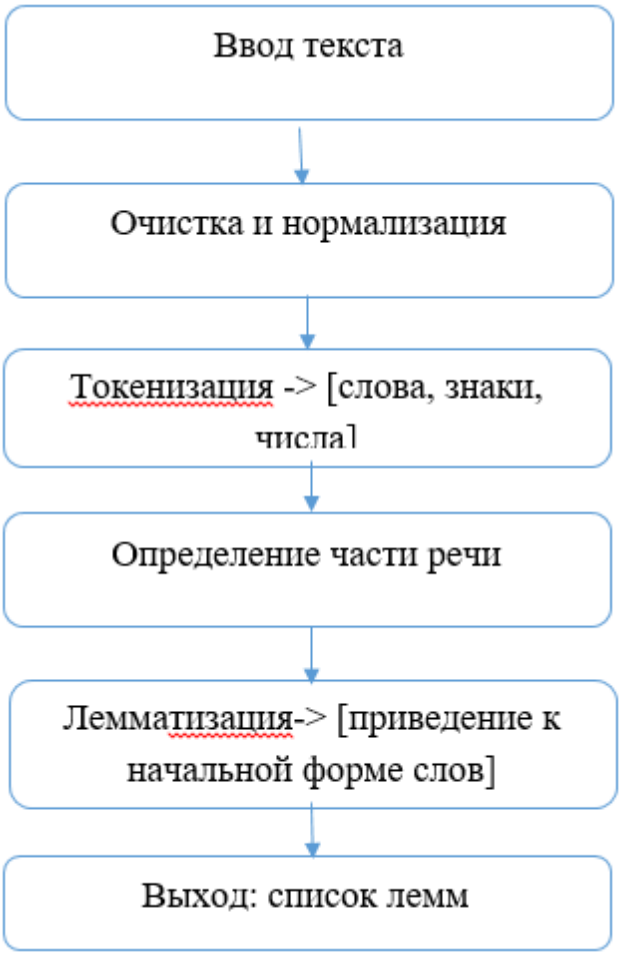
Минимальное расстояние Левенштейна между "mamlakatimizda" и "mustaqillik" равно 7. Это означает, что для превращения первой строки во вторую нужно минимум 7 операций удаления, вставки или замены символов.

При лемматизации, напрямую расстояние Левенштейна может не применяться, но если необходимо автокоррекция и исправление ошибок или же необходим выбор наиболее похожего правильного слова перед лемматизацией,

также при лемматизация низкокачественного текста может использоваться расстояние Левенштейна.

Приведём алгоритм лемматизации состоящий из следующих шагов. На первом шаге на вход подаются список токенов. На втором шаге определяется часть речи существительный, глагол, прилагательное, с использованием морфологического анализатора. На третьем шаге происходит поиск начальной формы слова, процесс приведения слова к его переводу или словарной форме, называемой леммой. Этот процесс важен при обработке естественного языка, он позволяет применять разные формы одного слова к единому представлению, что плодотворно влияет на качество анализа текста, поисковых систем, других задач НЛП [7]. С помощью нормализации текста можно установить смысловую связь между словами и упростить обработку и анализ текстов. На четвертом этапе токен заменяется на его лемму. На самом последнем этапе мы получаем список лемм.

Схема1. Итоговая схема алгоритма лемматизации.



References:

1. Evgeniy Gabrilovich, Shaul Markovitch. Feature generation for text categorization using world knowledge, 2005.

2. Rakhmanov Askar, Iskhakova Nargiza, Abduvalieva Zebiniso Word representation in vector space using word2vec model. Eurasian journal of mathematical theory and computer sciences Innovative Academy Research Support Center IF ,7.906.2025 C.54-59.
3. Raxmanov A.T., Abduvalieva Z.A. Application of the bag of words (BoW) model in natural language processing tasks. “Digital transformation: a new era in information technology, artificial intelligence and the economy” materials of the international scientific-practical conference april 16-17, C.37-41.2025
4. Rakhmanov Askar, Abduvalieva Zebiniso. Classification of text data. international scientific-electronic journal “pioneering studies and theories”. Vol. 1 No. 4 (2025)
5. Lafferty J., McCallum A., Pereira F.C.N. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. ICML, 2001
6. Automatic Lemmatization of Old English Class III Strong Verbs (L-Y) with ALOEV3, JOURNAL OF ENGLISH STUDIES – vol. 20 (2022): 237-266.
7. Improving Lemmatization of Non-Standard Languages with Joint Learning Enrique Manjavacas1 , Akos K ´ ad´ ar´ 2 , and Mike Kestemont1. pages 1493–1503 Minneapolis, Minnesota, June 2 - June 7, 2019