ISSN 2409-2665
Journal of Logistics, Informatics and Service Science
Vol. 12 (2025) No. 1, pp. 391-409
DOI:10.33168/JLISS.2025.0120

Arabic Part-of-Speech Tagging with Decision Tree: Effects of Algorithm and Dataset Size

Yousef Abu Elbeh[1], Maryam Alzawahreh[2], Shihadeh Alqrainy[1], Hasan Muaidi Alserhan[1]

[1] Al-Balqa Applied University, Jordan

[2] Hashemite University, Jordan

abuebeh@gmail.com, Maryam_alz@hu.edu.jo, alqrainy@bau.edu.jo, alserhan@bau.edu.jo

Abstract. This paper introduces the Arabic Decision Tree Tagger (ADTT), an innovative Machine Learning Part-of-Speech tagging system. Its primary function is constructing a tree using a training set, assigning the appropriate tag to each word in an untagged, raw, unvocalized Arabic corpus (the testing set), and generating a POS-tagged corpus. The decision tree approach used here incorporates two algorithms: ID3 and C4.5, both created by Ross Quinlan. These algorithms build decision trees from datasets for classification purposes. The distinction lies in C4.5, a more recent version, incorporating several features absent in ID3. These algorithms are designed to create a tree from the training set and utilize it to produce a POS-tagged corpus from the raw, untagged, unvocalized corpus. The Al-Watan corpus was used to establish both training and testing sets. The tagger underwent training eight times, with accuracy evaluated during testing. The best accuracy recorded was when the training set had 47965 words, leading to an accuracy of 75.88% with ID3 and 84.06% with C4.5.

Keywords: Tree Tagger, Part-of-Speech, Decision Tree, Token, Word Class.

1. Introduction

As a branch of Artificial Intelligence (AI), Natural Language Processing (NLP) is a field that grapples with the complexity and nuance of human languages. It focuses on analyzing, understanding, and generating human languages to facilitate interaction with computers in both written and spoken contexts. NLP is not a simple task for computers, as it requires an understanding of everyday human languages (e.g., English, Arabic, French) instead of programming languages (e.g., Java, C++, etc.). A computer that lacks human-like knowledge of the world and an understanding of phonetic structures finds it difficult to comprehend human languages (Alqrainy et al., 2008).

According to (Baker et al., 1994), data is the cornerstone of resolving potential uncertainties in NLP. A word, a phrase, or a sentence is considered ambiguous in NLP if it can be understood in multiple ways. This ambiguity, as noted by (Gazadr, 1990), is the most significant issue in NLP. At every level of language comprehension, natural language is rife with ambiguities, including lexical (words with multiple meanings), syntactic (words with different structural roles in a sentence), and semantic (sentences with multiple interpretations) ambiguities (Ceccato et al., 2004).

In addition, (Alqrainy, 2008) stated that the NLP field's primary objective is to resolve the ambiguity in human language. Whether in lexical, syntactic, or semantic form, the disambiguation process is the core initial step in most NLP tasks, such as machine translation, information retrieval, or similar tasks.

Part-of-speech (POS) tagging is the most common disambiguation process and has attracted significant interest from the NLP research community. POS tagging involves labeling or classifying each word in a written text according to its part of speech, such as noun, verb, preposition, adjective, etc. It addresses the resolution of lexical ambiguity.

A POS tagging system's primary challenge and ultimate goal is to resolve these lexical ambiguities. Lexical data encompasses not only the part of speech of a word but also its inflectional characteristics, such as tense, person, number, mood, case, and gender. In most instances, this data must be accessible to the tagging system. It is encoded in an explanatory symbol known as a tag and stored in a lexicon or dictionary (Halteren, 1999).

POS tagging is a crucial intermediate step in developing various NLP applications, including text-to-speech synthesis, speech recognition, information retrieval (IR), spelling correction, and parsing systems. Corpus linguistics is the most prominent and advanced field where POS tagging is employed (Halteren, 1999)(Alian & Awajan, 2018).

This paper presents a Machine Learning Part-of-Speech tagging system called Arabic Decision Tree Tagger (ADTT). The main function of the ADTT system is to build a tree from the training set and to assign the correct tag to each word in an untagged raw non-vocalized Arabic corpus which is a testing set, and to produce a POS tagged corpus. Decision Tree technique with two algorithms were used in this work, the ID3 algorithm and C4.5 algorithms. Al-Watan corpus is used to create a training set and testing set, which is a free corpus. The tagger was trained eight times, and accuracy was checked each time when testing the tagger. The highest accuracy was achieved when the training set volume contained 47965 words, which is 75.88% by ID3 and 84.06% by C4.5.

The current paper poses these two questions and provides proven answers:

1. Can decision tree algorithms be utilized to create an Arabic POS tagger using machine learning techniques?

2. Do decision tree algorithms yield satisfactory accuracy with Arabic text?

2. Related Work

(ALGahtani et al., 2009) developed a tagger based on Transformation-based learning (TBL) for tagging a modern standard Arabic text (MSA) and used a segment level on corpus rather than word level to produce a rule for retagging. (ALShamsi, et al., 2006) built a tagger based on a statistical language

model using Hidden Markov Model (HMM). (Diab, et al., 2004) displayed a machine learning approach using a support vector machine to find a solution for automatically annotating Arabic text with tags. (AL-Taani & AL-Rub, 2009) developed a tagging system based on a rule-based system for non-vocalized Arabic text to find the true tag for each word. (Ben Ali & Jarray, 2013) presented a part of speech tagging for the Arabic language based on a genetic algorithm (GA). (Yousif & Sembok, 2008) presented a tagger for the Arabic language based on the support vector machine with the Arabic tag set which contains 131 tags and a corpus containing 50000 words. (EL Hadj et al., 2009) presented tagger for Arabic based on sentence structure which combined the HMM with morphological analysis. (Mohamed & Kubler, 2010) compared two novel methods for Arabic part of speech tagging with a full tag set from Penn Arabic Treebank. The first approach did not use word segmentation because the full words used had complex tags. The other approach is based on segmentation which splits the word into segments then passes the segments to POS tagger then assigns a tag for each word. (Khoja, 2001) developed a tagger for Arabic text based on a hybrid technique which is a combination of rule-based and statistical technique. (Habash & Rambow, 2005) used a morphological analyzer for morphological disambiguation, tokenization, and part of speech tagging. (Marsi et al., 2005) developed an application for morphological analysis and part of speech tagging for the Arabic language by utilizing memory based learning with data originating from Arabic Treebank. (Kanaan, et al., 2013) built a fully automatic tagging system for Arabic text, which used vowelized text from the holy Qur'an and 242 non-vowelized abstracts that were chosen randomly from the Saudi Arabia National Computer Conference as input; the system contains a rule to find a tag for each word. The system achieved an accuracy of 93%. (Alqrainy et al., 2008) presented a new algorithm to give a word the right POS tag which belongs to a noun or verb group in the Arabic text. In addition, the authors presented ARABTAGS tagset. (Guiassa, 2006) presented a hybrid method, which combines rule-based and machine learning for tagging Arabic text. (Alian & Awajan, 2018) provide 13 types of Ar-tags. The tags are based on the Arabic language or English language; some tags are utilized for the Arabic language.

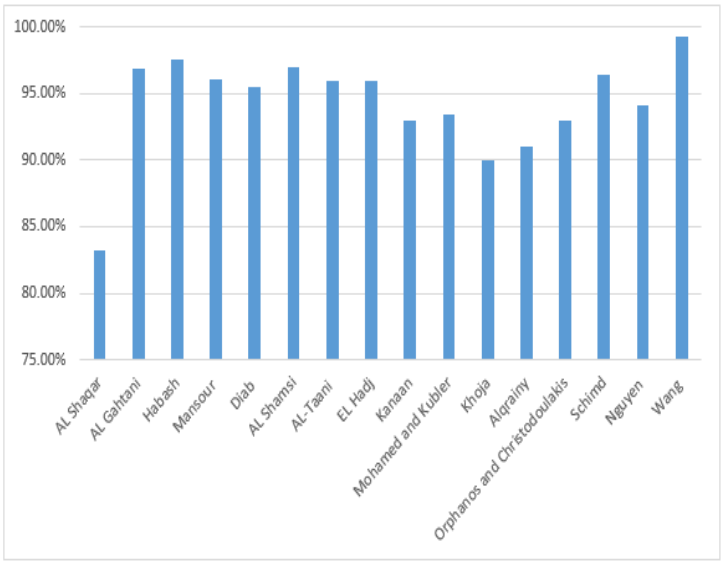


Fig.1: Overview of Accuracy in Various Arabic POS Tagging Approaches.

3. Decision Tree Techniques

A method for approximating discrete-valued target functions is called decision tree learning, where a decision tree represents the learned function as a set of if-then rules to enhance human readability. According to (Mitchell, 1997), one of the most popular inductive inference algorithms has been applied

to diagnose medical cases and assess loan applicants' credit risk. Researchers have developed enhancements to improve performance and the ability to handle various types of data within decision tree algorithms. (Patel & Rana, 2011) discuss several important algorithms. The description is provided below:

3.1. ID3 Algorithm

ID3 (Iterative Dichotomiser 3) decision tree algorithm is developed by Quinlan. The information gain approach is generally used to determine suitable properties for each node of a generated decision tree in the decision tree method. The attribute of the highest information gain can be selected as the test attribute of the current node and the smallest partitioned is the classification of the training sample. This property reduces the mixture degree of different types for all generated sample subsets to a minimum. According to (Patel & Rana, 2011), the dividing number of object classifications is reduced effectively using the information theory approach.

ID3 used an information gain measure to find the best attribute at each node for testing. The attribute that returns the highest value from the Information Gain expression is taken to test in the node.

$$InformationGain(S, A) = Entropy(S) - \sum_{v \in Value(A)} \frac{|S_v|}{|S|} . Entropy(S_v)$$

Where:

$$Entropy(S) = \sum_{i=1}^{n} -p_i log_2(p_i)$$

Where:

C is an arbitrary categorization into c1,…....,cn.

S is a set of examples.

Pi is a proportion of examples in ci.

A is an attribute.

V is a value for attribute A.

Below is the pseudocode for the ID3 algorithm (Patel & Rana, 2011):

ID3 (Examples, Target Attribute, Attributes)

Create a root node for the tree

If all examples are positive, Return the single-node tree Root, with label = +.

If all examples are negative, Return the single-node tree Root, with label = -.

If number of predicting attributes is empty, then Return the single node tree Root, with label = most common value of the target attribute in the examples.

Otherwise Begin End

A = The Attribute that best classifies examples.

Decision Tree attribute for Root = A.

For each possible value, vi, of A,

Add a new tree branch below Root, corresponding to the test A = vi

Let Examples vi be the subset of examples that have vi for A

If Examples vi is empty

Then below this new branch add a leaf node with label = most common target value in the examples

Else below this new branch add the subtree

ID3 (Examples vi, Target Attribute, Attributes– {A})

Return Root

3.2. C4.5 Algorithm

C4.5 is an algorithm used to generate a decision tree developed by Ross Quinlan, an extension of Quinlan's earlier ID3 algorithm. C4.5 is often referred to as a statistical classifier that can be used for classification. The information gain by the algorithm is used as splitting criteria that accepts data with categorical or numerical values. The C4.5 can easily handle missing values according to (Patel & Rana, 2011), to gain calculations the missing attribute values are not utilized.

C4.5 used an information gain measure to find the best attribute at each node for testing. The attribute that returns the highest value from the information gain expression is taken to test in the node. Below is the pseudocode for the C4.5 algorithm (Patel & Rana, 2011):

Input: an attribute-valued dataset D

Tree = {}

If D is "Pure" OR other stopping criteria met, then

Terminate

End if

For all attribute α ϵD do

Compute information-theoretic criteria if we split on α

End for

αbest = Best attribute according to above computed criteria

Tree = Create a decision node that tests αbest in the root

Dv = induced sub-datasets from D based on αbest

For all Dv do

Tree v = C4.5(Dv)

Attach Tree v to the corresponding branch of tree

End for

Return Tree

4. A Description of the Tagger System

The main function of ADTT is to take an untagged non-vocalized Arabic text as input, and to produce

a POS tagged unvocalized Arabic corpus. ADTT as shown in Figure 2 is composed of two main modules: Tokenizer Module, and Tree Module.

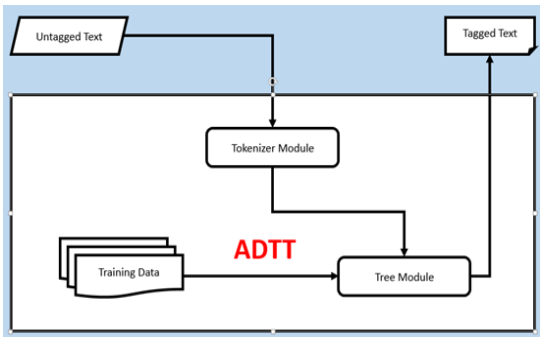


Fig. 2: An overview of ADTT

The following list provides a detailed explanation of each module's function:

Tokenizer Module

A token is defined as a character in a sequence that creates a collective meaning. Special characters, numbers, and words are represented by tokens. The aim of a tokenizer module is to take input text, which is untagged text, and convert it into a form that can be easily managed by the machine. This conversion is called tokenization. Finding and identifying words, numbers, punctuation, and other marks characters by using the delimiter, which is the space, is the responsibility of the tokenization process (Alqrainy, 2008).

Tree Module

The primary function of this module is to input each testing word into a decision tree and return each testing word with its associated tag. It utilizes the C4.5 algorithm or the ID3 algorithm and checks the features of each word through internal nodes until it reaches a leaf node that contains the tag. The decision tree is built using a training set, starting from the top at the root node and extending down to the bottom where the leaf nodes are located.

5. Tagging Process

ADTT performs many steps during the tagging process. Below are the steps of our technique's tagging process. After implementing the tokenization process on untagged text, the extract features function is used to find features for the next process. Table 1 shows the characteristics that should be extracted.

Step - 1:

The first step is to take untagged text and split it into tokens (words) using a tokenizer and remove digital characters (0 -9) in addition to non-Arabic text (e.g., English …etc.).

Step -2:

The second step involves extracting characteristics from each word to use as a check in the internal node. Table 1 displays the token characteristics utilized in this paper.

Table 1: Token’s Characteristics

Characteristics	Description
Character one	First character in token
Character two	Second character in token
Character n-1	One character before last character in token
Previous word tag	Any tag from tag set
Count of Word Character	Number of character in token

Step -3:

Input token characteristics into the decision tree constructed from training data and evaluate features at internal nodes.

Step -4:

Assign a detailed tag to each token from the tag set. Table 2 provides an overview of general tags from ARABTAGS.

Table 2: General Tags from ARABTAGS

Tag	**Description**	**Tag**	**Description**
VPr	Perfect verb	NAd	Adverb
VPi	Imperfect verb	NNm	Numeral noun
VPm	Imperative verb	Pp	Preposition
NPn	Proper noun	NIn	Interrogative noun
NCm	Common noun	PFt	Future particle
NRe	Relative noun	PEm	Emphasis particle
NIs	Instrument noun	PVo	Vocative particle
NNp	Noun of place	NCo	Conjunction particle
NNt	Noun of time	NAn	Annulment particle
NPs	Pronoun	NEx	Exception particle
NCj	Conjunctive noun	NSb	Subjunctive particle
NCn	Condition noun	PVr	Verbal noun particle
NDe	Demonstrative noun	PEp	Explanation particle

This paper employs the ID3 and C4.5 algorithms as the primary methods for tagging Arabic words, given that they are the most widely used decision-making techniques. The subsequent example elaborates on these techniques in greater detail. Table 3 presents a snapshot of the training data utilized in the examples.

Table 3: Snapshot of the Training Data

Word	First Char	Second Char	Last Char - 1	Last Char	Count of Char	Previous Tag	Tag
المعلم	ا	ل	ل	م	6	VePeMaSnTh	NuAjMaSn
يعملن	ي	ع	ل	ن	5	NuAjFePl	VePeFiPlTh
في	ف	ي	ف	ي	2	NuAjMaSn	PrPp
الى	ا	ل	ل	ى	3	NuAjFePl	PrPp
المعلمات	ا	ل	ا	ت	8	VePeFeSnTh	NuAjFePl
المعلمون	ا	ل	و	ن	8	VePeMeSnTh	NuAjMaPl
يشربن	ي	ش	ب	ن	5	NuAjFePl	VePeFeThPlTh
مساعدات	م	س	ا	ت	5	VePeMeSnTh	NuAjFePl
على	ع	ل	ل	ى	3	NuAjFePl	PrPp
مسعفون	م	س	و	ن	6	VePeMaSnTh	NuAjMaPl

The following steps demonstrate how the decision tree was constructed using the ID3 and C4.5 algorithms:

ID3

- **Step – 1:**

Find the Entropy for a target which is a tag using the following equation:

$$Entropy(S) = \sum_{i=1}^{n} -p_i log_2(p_i)$$

P (NuAjMaSn) = 1/10 = 0.1

P (NuAjFePl) = 2/10 = 0.2

P (NuAjMaPl) = 2/10 = 0.2

P (VePiFePlTh) = 2/10 = 0.2

P (PrPp) = 3/10 = 0.3

$$Entropy(Tag) = -0.1\ log_2(0.1) - 0.2\ log_2(0.2) - 0.2\ log_2(0.2) - 0.2\ log_2(0.2) - 0.3\ log_2(0.3)$$
$$= 2.25$$

- **Step – 2:**

Find information gain for characteristic using:

$$InformationGain(S, A) = Entropy(S) - \sum_{v \in Value(A)} \frac{|S_v|}{|S|} . Entropy(S_v)$$

$$Entropy(ا) = -\frac{1}{4}log_2\left(\frac{1}{4}\right) - \frac{1}{4}log_2\left(\frac{1}{4}\right) - \frac{1}{4}log_2\left(\frac{1}{4}\right) - \frac{1}{4}log_2\left(\frac{1}{4}\right) = 2$$

$$Entropy(ي) = -\frac{2}{2}log_2\left(\frac{2}{2}\right) = 0$$

$$Entropy(ف) = -\frac{1}{1}log_2\left(\frac{1}{1}\right) = 0$$

$$Entropy(م) = -\frac{1}{2}log_2\left(\frac{1}{2}\right) - \frac{1}{2}log_2\left(\frac{1}{2}\right) = 1$$

$$Entropy(ع) = -\frac{1}{1}log_2\left(\frac{1}{1}\right) = 0$$

$$InformationGain(First\ char) = 2.25 - \left(\frac{4}{10} * 2 + \frac{2}{10} * 0 + \frac{1}{10} * 0 + \frac{2}{10} * 1 + \frac{1}{10} * 0\right)$$

$$= 1.25$$

- **Step – 3:**

Do step 2 for each characteristic then choose best one to use it as a checker in internal node:

$$InformationGain(Second\ char) = 1.09$$

$$InformationGain(Last\ char - 1) = 1.65$$

$$InformationGain(Last\ char) = 1.85$$

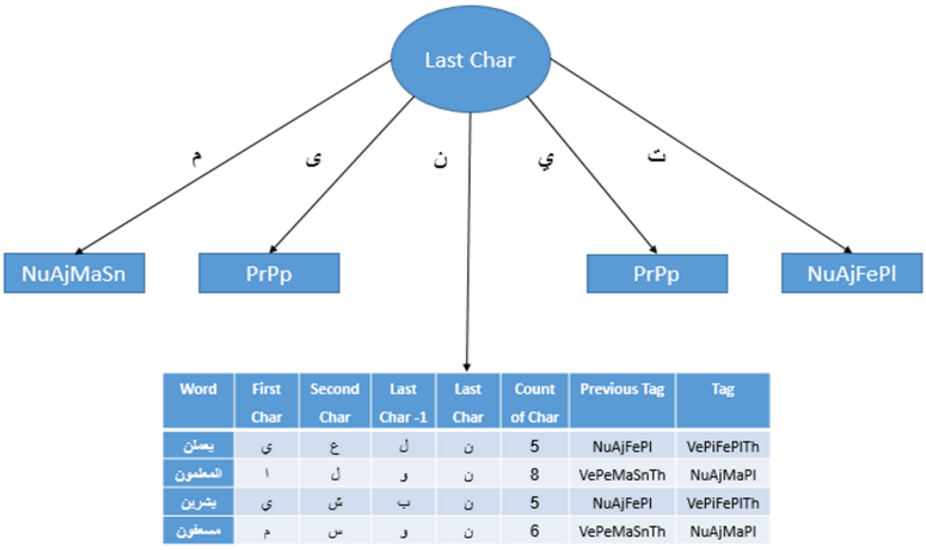
$$InformationGain(Count\ of\ char) = 1.85$$

$$InformationGain(Previous\ word\ tag) = 1.25$$

The best characteristic is Last Char so the decision tree will begin with this as root node. Figure 3 display the first version of tree after choosing the first checker node.

- **Step – 4:**

Repeat steps 2 and 3 for each non leaf node and stop till use all characteristics or all branches in the tree forward to leaf nodes.



Word	First Char	Second Char	Last Char -1	Last Char	Count of Char	Previous Tag	Tag
يعلن	ى	ع	ل	ن	5	NuAjFePl	VePiFePlTh
المعلمون	ا	ل	و	ن	8	VePeMaSnTh	NuAjMaPl
يشرين	ى	ش	ب	ن	5	NuAjFePl	VePiFePlTh
مسعفون	م	س	و	ن	6	VePeMaSnTh	NuAjMaPl

Fig. 3: First Version of ID3 Tree

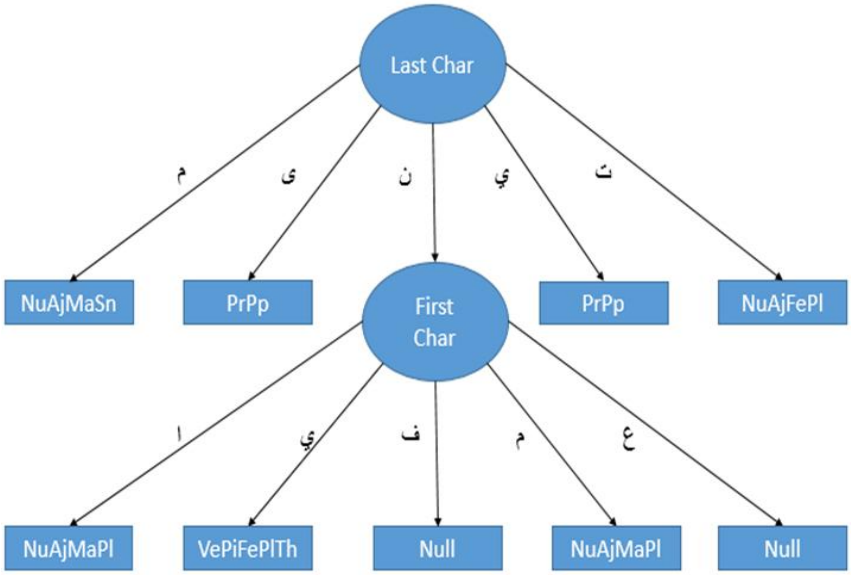


Fig. 4: Full ID3Ttree

Figure 4 displays the full tree for ID3.

C4.5 Algorithm

The C4.5 algorithm closely resembles the ID3 algorithm. It incorporates several enhancements, including the capability to manage continuous attributes and prune after generating a tree. The main steps of the C4.5 algorithm are as follows:

- **Step – 1:**

Find Entropy for a target that is tagged using the following equation:

$$Entropy(S) = \sum_{i=1}^{n} -p_i log_2(p_i)$$

P(NuAjMsSn) = 1/10 = 0.1

P(NuAjFePl) = 2/10 = 0.2

P(NuAjMsPl) = 2/10 = 0.2

P(VePiFePlTh) = 2/10 = 0.2

P(PrPp) = 3/10 = 0.3

$Entropy(Tag) = -0.1\ log_2(0.1) - 0.2\ log_2(0.2) - 0.2\ log_2(0.2) - 0.2\ log_2(0.2) - 0.3\ log_2(0.3) = 2.25$

- **Step – 2:**

Find information gain for characteristic using:

$$InformationGain(S, A) = Entropy(S) - \sum_{v \in Value(A)} \frac{|S_v|}{|S|} . Entropy(S_v)$$

$$Entropy(ا) = -\frac{1}{4}log_2\left(\frac{1}{4}\right) - \frac{1}{4}log_2\left(\frac{1}{4}\right) - \frac{1}{4}log_2\left(\frac{1}{4}\right) - \frac{1}{4}log_2\left(\frac{1}{4}\right) = 2$$

$$Entropy(ي) = -\frac{2}{2}log_2\left(\frac{2}{2}\right) = 0$$

$$Entropy(ف) = -\frac{1}{1}log_2\left(\frac{1}{1}\right) = 0$$

$$Entropy(م) = -\frac{1}{2}log_2\left(\frac{1}{2}\right) - \frac{1}{2}log_2\left(\frac{1}{2}\right) = 1$$

$$Entropy(ع) = -\frac{1}{1}log_2\left(\frac{1}{1}\right) = 0$$

$$InformationGain(First\ char) = 2.25 - \left(\frac{4}{10} * 2 + \frac{2}{10} * 0 + \frac{1}{10} * 0 + \frac{2}{10} * 1 + \frac{1}{10} * 0\right)$$
$$= 1.25$$

- **Step – 3:**

Complete step 2 for each characteristic, then select the best one to use as a checker in the internal node:

$$InformationGain(Second\ char) = 1.09$$

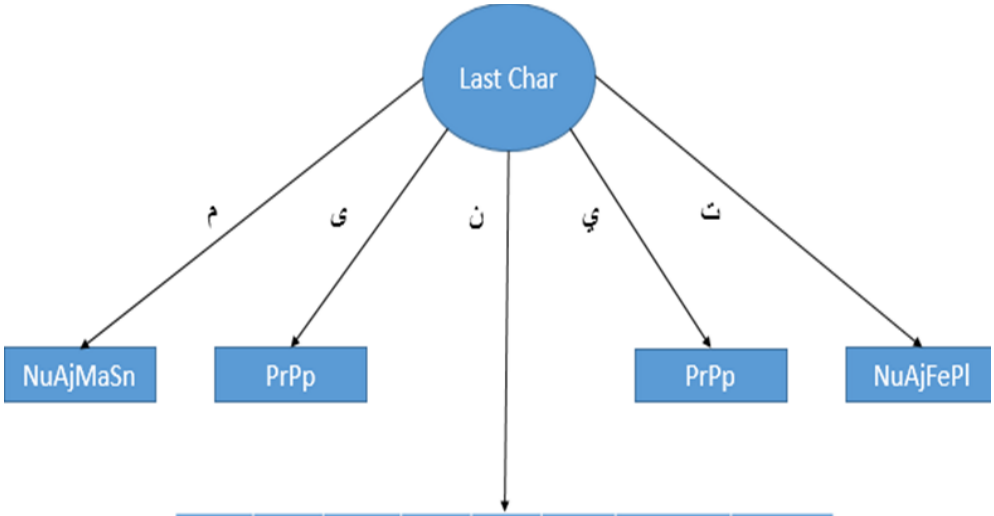
$$InformationGain(Last\ char - 1) = 1.65$$

$$InformationGain(Last\ char) = 1.95$$

$$InformationGain(Count\ of\ char) = 1.00$$

$$InformationGain(Previous\ word\ tag) = 1.25$$

The best characteristic is the Count of Char so the decision tree will begin with this as the root node. Figure 5 displays the first version of the tree after choosing the first checker node.



Word	First Char	Second Char	Last Char -1	Last Char	Count of Char	Previous Tag	Tag
يعلن	ى	ع	ل	ن	5	NuAjFePl	VePiFePlTh
المعلمون	ا	ل	و	ن	8	VePeMaSnTh	NuAjMaPl
يشرين	ى	ش	ب	ن	5	NuAjFePl	VePiFePlTh
مسعفون	م	س	و	ن	6	VePeMaSnTh	NuAjMaPl

Fig. 5: First Version of C4.5 Tree

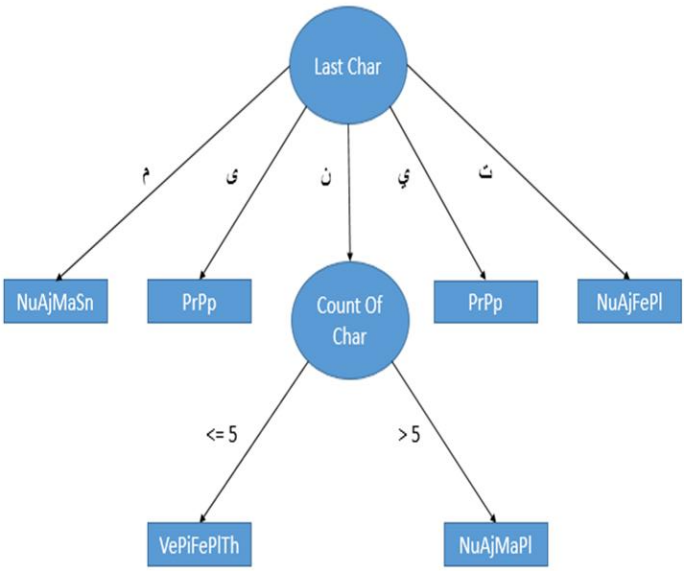


Fig. 6: Full Version of C4.5 Tree

Figure 6 displays full tree for C45.5.

6. Experimental Result

6.1. Training DataSet

Unvocalized Arabic texts are collected from a free Arabic corpus available for download (Al-Watan, 2004) to train the tagger. This corpus contains over 18,000 articles covering various subjects,

including sports, local and global affairs, cultural topics, and regional issues.

For the training dataset, we utilized 47965 words from various topics, extracted features for each word, and manually assigned a tag to each word by an Arabic linguist.

For the testing dataset, we used 4,718 words from the same corpus across various topics. The tagger extracted characteristics for each word, and the results were reviewed by the same Arabic linguist who evaluated the training set. The tagger underwent multiple tests, depending on the size of the training set, as shown in Table 4.

Table 4: Sets Used for Training

Set No.	No. of Words	No. of Articles
1	5458	10
2	12163	20
3	18856	30
4	23210	40
5	29361	50
6	36080	60
7	44650	70
8	47965	85

6.2. Tagger Experiments and Accuracy Measurement

Eight experiments were conducted to evaluate the tagger system. These experiments were carried out on sets 1 through 8 to train the ID3 algorithm in the tagger and test the algorithm after each set with 4,718 words. The same eight sets were also utilized to train C4.5 and test the algorithm after each set with 4,718 words.

More than one measurement can be used to assess the performance of tagger systems. Success rate, ambiguity, recall, and precision are the most commonly used measures for determining the accuracy of tagger output. In this work, the success rate measure is utilized. It is expressed as a percentage and defined as follows:

$$Success\ rate = \frac{Number\ of\ coreectly\ tagged\ tokens}{total\ number\ of\ tokens} * 100\%$$

6.2.1 Experiment 1

The first experiment involved using Set 1 to train the tagger and testing it with 4,718 words. The tagger achieved an accuracy of 68.57% with C4.5 and 45.36% with ID3 on the test words, as shown in Table 5.

Table 5: Success Ratios for Experiment 1

Article #	No. of Words	C4.5		ID3	
		Correct Tags	Accuracy	Correct Tags	Accuracy
1	506	346	68.38%	252	49.8%
2	121	84	69.42%	46	38.02%
3	248	183	73.79%	134	54.03%

4	464	310	66.81%	205	44.18%
5	116	76	65.52%	53	45.69%
6	152	102	67.11%	74	48.68%
7	1107	687	62.06%	474	42.82%
8	1208	859	71.11%	516	42.72%
9	796	588	73.87%	386	48.49%
Total	4718	3235	68.57%	2140	45.36%

6.2.2 Experiment 2

The second experiment was performed using the set 2 to train the tagger and 4718 words to test the tagger. Tagger correctly tagged 74.80% by C4.5, and 57.88% by ID3 from testing words as shown in Table 6.

Table 6: Success Ratios for Experiment 2

Article #	No. of Words	C4.5		ID3	
		Correct Tags	Accuracy	Correct Tags	Accuracy
1	506	374	73.91%	312	61.66%
2	121	89	73.55%	76	62.81%
3	248	186	75.55%	140	56.45%
4	464	336	72.41%	263	56.68%
5	116	88	75.86%	51	43.97%
6	152	122	80.26%	107	70.49%
7	1107	794	71.73%	588	53.12%
8	1208	926	76.66%	689	57.00%
9	796	614	77.14%	505	63.44%
Total	4718	3529	74.80%	2731	57.88%

6.2.3 Experiment 3

The third experiment utilized set 3 to train the tagger, testing it with 4,718 words. The tagger correctly tagged 79.14% of the testing words using C4.5 and 66.36% with ID3, as shown in Table 7.

Table 7: Success Ratios for Experiment 3

Article #	No. of Words	C4.5		ID3	
		Correct Tags	Accuracy	Correct Tags	Accuracy
1	506	381	75.30%	335	66.21%
2	121	93	76.86%	79	65.29%
3	248	187	75.40%	151	60.89%
4	464	358	77.16%	301	63.87%
5	116	90	77.59%	66	56.90%

6	152	135	88.82%	123	80.92%
7	1107	865	78.82%	739	66.76%
8	1208	958	79.30%	764	63.25%
9	796	667	83.79%	573	71.98%
Total	4718	3734	79.14%	3131	66.36%

6.2.4 Experiment 4

The fourth experiment utilized set 4 to train the tagger and 4,718 words to test it. The tagger achieved a correctness rate of 80.54% using C4.5 and 68.93% using ID3 on the tested words, as shown in Table 8.

Table 8: Success Ratios for Experiment 4

Article #	No. of Words	C4.5		ID3	
		Correct Tags	Accuracy	Correct Tags	Accuracy
1	506	399	78.85%	346	68.38%
2	121	94	77.69%	85	70.25%
3	248	188	75.81%	154	62.10%
4	464	363	78.23%	320	68.97%
5	116	91	78.45%	70	60.34%
6	152	139	91.45%	131	86.18%
7	1107	883	79.77%	744	67.21%
8	1208	970	80.30%	815	67.47%
9	796	673	84.55%	587	73.74%
Total	4718	3800	80.54%	3252	68.93%

6.2.5 Experiment 5

The fifth experiment utilized set 5 to train the tagger with 4,718 words for testing. The tagger correctly tagged 82.56% using C4.5 and 73.71% with ID3, as shown in Table 9.

Table 9: Success Ratios for Experiment 5

Article #	No. of Words	C4.5		ID3	
		Correct Tags	Accuracy	Correct Tags	Accuracy
1	506	401	79.25%	342	67.59%
2	121	99	81.82%	110	90.91%
3	248	198	79.84%	155	62.50%
4	464	373	80.39%	331	71.34%
5	116	91	78.45%	78	67.24%
6	152	143	94.08%	141	92.76%
7	1107	906	81.84%	760	68.65%
8	1208	983	81.37%	810	67.07%

9	796	685	86.06%	600	75.38%
Total	4718	3879	82.56%	3327	73.71%

6.2.6 Experiment 6

The sixth experiment involved using set 6 to train the tagger and 4,718 words to evaluate it. The tagger achieved an accuracy of 82.59% with C4.5 and 72.99% with ID3 on the test words, as shown in Table 10.

Table 10: Success Ratios for Experiment 6

Article #	No. of Words	C4.5		ID3	
		Correct Tags	Accuracy	Correct Tags	Accuracy
1	506	415	82.02%	371	73.32%
2	121	102	84.30%	109	90.08%
3	248	200	80.65%	157	63.31%
4	464	370	79.74%	344	74.14%
5	116	97	84.26%	87	75.00%
6	152	143	94.08%	142	93.42%
7	1107	898	81.12%	750	67.75%
8	1208	999	82.70%	854	69.59%
9	796	690	86.68%	630	79.15%
Total	4718	3914	82.95%	3444	72.99%

6.2.7 Experiment 7

The seventh experiment used set 7 to train the tagger and 4,718 words to test it. The tagger achieved an accuracy of 83.55% with C4.5 and 74.95% with ID3 on the testing words, as shown in Table 11.

Table 11: Success Ratios for Experiment 7

Article #	No. of Words	C4.5		ID3	
		Correct Tags	Accuracy	Correct Tags	Accuracy
1	506	403	79.64%	379	74.90%
2	121	107	88.43%	110	90.91%
3	248	205	82.66%	166	66.94%
4	464	363	78.23%	347	74.78%
5	116	95	81.90%	86	74.14%
6	152	142	93.42%	139	91.45%
7	1107	915	82.66%	778	70.28%
8	1208	1012	83.77%	884	73.18%
9	796	700	87.94%	647	81.28%
Total	4718	3942	83.55%	3536	74.95%

6.2.8 Experiment 8

The eighth experiment utilized set 8 to train the tagger and 4,718 words to test it. The tagger achieved an accuracy of 84.06% with C4.5 and 75.88% with ID3, as shown in Table 12.

Table 12: Success Ratios for Experiment 8

Article #	No. of Words	C4.5		ID3	
		Correct Tags	Accuracy	Correct Tags	Accuracy
1	506	417	82.14%	380	75.10%
2	121	106	89.26%	112	92.56%
3	248	206	83.06%	170	68.55%
4	464	363	78.23%	349	75.22%
5	116	95	81.90%	90	77.59%
6	152	142	93.42%	139	91.45%
7	1107	921	83.20%	795	71.82%
8	1208	1020	84.44%	891	73.76%
9	796	696	87.44%	654	82.16%
Total	4718	3966	84.06%	3580	75.88%

6.3. Experimental Results Analysis

The overall results of the eight experiments are summarized in Figure 7.

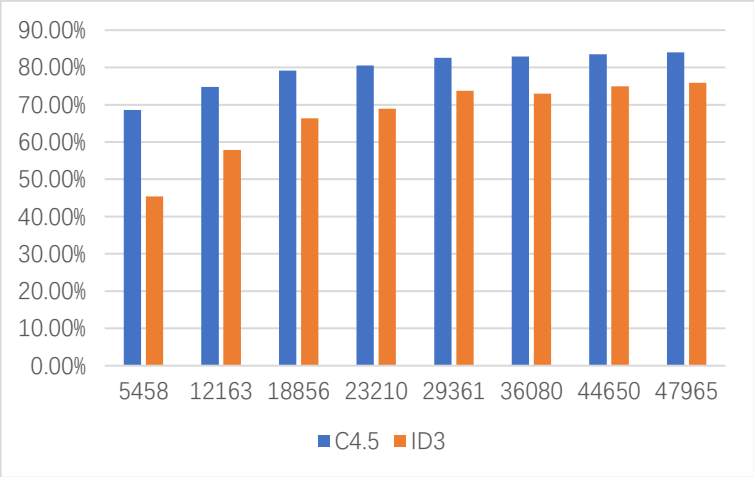


Fig. 7: Overall Results

The accuracy of the results is not guaranteed to be as precise as expected. The following factors may have contributed to this:

1- Training Volume:

Since machine learning techniques need a large training set, results are validated when accuracy improves by including more tokens in the training set.

2- No Stemming:

Stemming is the process of reducing a word to its root by eliminating all prefixes, suffixes, and infixes. The number of classes in the module is high because stemming is not used in this work. For example, the word وبهم (wbhm) has three tokens: و (w), ب (b), and هم (hm), which correspond to the PCo, PPp, and NPsPlThMa tags, respectively.

3- Characteristics:

In this study, certain characteristics in the training dataset are influenced by prior token outputs. Consequently, if a token is incorrectly tagged in the testing dataset, the subsequent token's classification may also be inaccurate due to flawed input.

4- Unvocalized text:

Due to the use of unvocalized Arabic text, a token may be classified with the incorrect tag. For example, the tag for the token كتب (he wrote) is VPrSnThMa, while the tag for the token كتب (books) is NCmPlFeb. These two tokens have the same input features, except for the type of the previous word, which varies based on the preceding token's output. Therefore, if these two tokens are entered in the tagger, there is a high probability that they will receive the same tag.

7. Conclusion

Many part-of-speech tagging systems based on grammar rules, statistics, or machine learning approaches have been developed with high tagging accuracy, primarily for English. For other languages, such as Arabic, some systems have been created for part-of-speech tagging. These systems are categorized into three types based on the input text: tagging vocalized text, tagging unvocalized text, and tagging partially vocalized text.

This paper introduces a machine learning tagging system for unvocalized Arabic text using two decision tree algorithms: C4.5 and ID3. The Al-Watan corpus has been utilized, which is divided into two sets: training and testing datasets.

ADTT is trained eight times with different sizes of training sets and tested each time with the same size of testing set. The system achieved maximum accuracy when using 47965 words for training. The result was 84.06% for C4.5 and 75.88% for ID3.

References

AlGahtani S., Black W. & McNaught J. Arabic Part-Of-Speech Tagging using Transformation-Based Learning. "Proceedings of the Second International Conference on Arabic Language Resources and Tools", Cairo, Egypt, 2009.

Alian, M., & Awajan, A.A. 2018. Arabic Tag Sets: Review. Intelligent Systems with Applications.

Alqrainy S., AlSerhan H., & Ayesh A. Pattern-Based Algorithm for Part-of-Speech Tagging Arabic Text. "2008 International Conference on Computer Engineering & Systems", Cairo, Egypt, 2008, pp. 119–124.

AlShamsi, F. & Guessoum, A. A hidden Markov model-based POS tagger for Arabic. "In: Proceeding of the 8th International Conference on the Statistical Analysis of Textual Data", France, 2006.

Al-Taani, A. and Al-Rub S. 2009. A rule-based approach for tagging non-vocalized Arabic words. Int. Arab J. Inf. Technol. Vol. 6, 320-328.

Al-Watan Arabic Corpus Files, 2004. https://sourceforge.net/projects/arabiccorpus/files/

Baker K.L., Frans A. & Jordan P.W. Coping with ambiguity in knowledge-based natural language analysis. "In the 8th International FLAIRS Conference", USA, 1994.

Ben Ali B. & Jarray F. 2013. Genetic approach for Arabic part of speech tagging. International Journal on Natural Language Computing. Vol. 2.

Ceccato M., Kiyavitskaya N., Zeni N., Mich L., & Berry D. Ambiguity identification and measurement in natural language texts. Technical Report DIT-04-111, Informatica e Telecomunicazioni, University of Trento., 2004.

Diab, M., Hacioglu, K., & Jurafsky, D. Automatic tagging of Arabic text: From raw text to base phrase chunks. "Proceedings of HLT-NAACL 2004 Short Papers. Association for Computational Linguistics", 2004.

El Hadj Y., Al-Sughayeir I. & Al-Ansari A. Arabic part-of-speech tagging using the sentence structure. "Proceedings of the Second International Conference on Arabic Language Resources and Tools", Cairo, Egypt, 2009.

Gazadr G. 1990. Natural Language Processing in LISP: An Introduction to Computational Linguistics. Addison-Wesley, Reading, MA.

Guiassa Y. T. & Tayeb L. M., 2006. Tagging by Combining Rules-based and Memory-based Learning. Information Technology Journal, Vol. 5, 679-684.

Habash N. & Rambow O. Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. "Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics ({ACL}`05", USA, 2005, pp. 573-580.

Halteren H.1999. Syntactic Wordclass Tagging, Vol. 9. Springer Dordrecht, Netherlands, pp. 207-216.

Kanan G., Al-Shalabi R. & Sawalha M. Full automatic Arabic text tagging system. "The proceedings of the International Conference on Information Technology and Natural Sciences", Amman, Jordan, 2003, pp. 258-267.

Khoja S. APT: Arabic part-of-speech tagger. "Proceedings of the Student Workshop at NAACL", 2001.

Marsi E., Bosch A., & Soudi A. Memory-Based Morphological Analysis Generation and Part-of-Speech Tagging of Arabic. "In Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages", USA, Michigan, 2008, pp. 1-5.

Mitchell T. M. (1997). Decision Tree Learning. In: Machine Learning, McGraw-Hill Education-Europe, 52-80.

Mohamed E. & Kubler S. Arabic Part of Speech Tagging. "Proceedings of the Seventh International Conference on Language Resources and Evaluation ({LREC}`10)", Malta, 2010.

Patel B. R. & Rana K. K. 2011. A Survey on Decision Tree Algorithm for Classification. International Journal of Engineering Development and Research. Vol. 2, 1-5.

Yousif J. H. & Sembok M. T. Arabic part-of-speech tagger based Support Vectors Machines. "2008 International Symposium on Information Technology", Kuala Lumpur, Malaysia, 2008, pp. 1-7.