UNIVERSITY OF NAIROBI
SCHOOL OF COMPUTING AND INFORMATICS

NAME ENTITY RECOGNITION AND PART OF SPEECH TAGGING: CASE STUDY OF KĪKAMBA

BY

KITUKU, BENSON NZIOKA
P58/70487/08

Supervisor
Dr. P. W. WAGACHA
April 2011

Submitted in partial fulfillment of the requirements of the Master of Science in Computer Science

DECLARATION:

This project, as presented in this report, is my original work and has not been presented for any other university award.

[illegible]
....
Signature
Kituku B.

June 16th 2011
....
Date:

This project has been submitted as partial fulfillment of requirements of the Master of Science in Computer Science of the University of Nairobi with my approval as the University Supervisor.

[illegible]
....
Signature
Dr. Wagacha

24th Aug 2011
....
Date:

ABSTRACT

There has been exponential multiplication of electronic information for the last two decades which has generated a large digital library for everyone to access over the internet. However, this library consists of unstructured documents where queries cannot be run as with a database so as to get preview of the content or certain details of interest. As a result, a need for language tool arises.

Natural language processing has provided a channel whereby the above challenge can be resolved using Name entity recognition (NER) in which a machine learning system is developed which can identify organization, personal and location names in various documents and report them from which you can get a glimpse of the contents of the documents.

In this project we present a Kĩkamba Name Entity Recognition using a memory based approach where supervised and bootstraps learning methods are applied to a carefully annotated corpus. To build the training set, a corpus is manually annotated. An annotated seed is also provided to facilitate bootstrap. Simultaneously, generation of Part of Speech tagging is done. The resultant classifiers are evaluated. The Aim of the project is a tool for analysis of electronic documents and at the same time find out the challenges that are peculiar to Kĩkamba language so as to compare with other languages which already have been tackled.

ACKNOWLEDGMENTS

My deepest thanks are to my parents Joseph Kituku and Priscilla Kanini for bringing me up well and teaching me that patience and hard work in God pays and finally, for financial assistance along the way. More so, for all the things parents deserve thanks for but so rarely get.

Many thanks to Dr Guy De Pauw for opening my mind in the field of Natural language processing (NLP), his inspiration, encouragement and his constant constructive criticism on various stages of developing this project.

Dr. Peter Waiganjo who has being my supervisor, Much appreciation for your patience and providing some Kĩkamba corpus and igniting my mind in terms of journal papers. I also appreciate the panelists Joseph Ogutu and Andrew Mwaura for their questions and criticism during every presentation.

Finally I appreciate Lydia Karauki, for proofing reading the whole document and enabling the completion of the dissertation on time.

Table of Contents

Table of figures

Table of tables

List of abbreviation

MUC	Message understanding conference
IE	Information extraction
I	Inside
O	Outside
B	Beginging
PoS	Part of speech tagging
I-PER/ B-PER	Person name
I-LOC/ B-LOC	Location name
I-ORG/B-ORG	Organization name
NER	Name entity regonization
NE	Name entity
NLP	Natural language processing
f	Function
d	Training examples
h	Hypothesis
Y	Desired output
K-nn	K nearest neighbor
MBL	Memory based lerner
IG	Information gain.
TIMBL	Tilburg Memory-Based Learner
X	Inputs
MBT	Memory based tagger

1.0 INTRODUCTION

The issue of Named Entity Recognition was one of the four themes of the Sixth Message Understanding Conference (MUC-6) (Grishman and Sundheim,1996). Although the focus was on defense related articles then, there has been a tremendous increase in research efforts over the years for different domains and languages, as presented in (Nadeau and Sekine, 2007).

During the conference an encoding called IOB tags arose, which indicates a word is outside of an entity (O), inside an entity (I) or at the beginning of an entity (B) For example in the sentence Benson Kituku uka vaa. As the associations tags B-PER, I-PER, O, O. whereby Benson start entity of type person and Kituku continues it while the rest are not entities. While Part of Speech tagging (PoS) is the process in which syntactic categories are assigned to words which also can be translated as mapping from sentences to strings of tags(Daelemans at el, 2001). The task (PoS) is of key usefulness to many subsequent manipulation of text, in that it provides a useful abstraction from the actual words if we want to process all words that belong to special class (get all verbs in a documents) and too provides superficial degree of disambiguation's in the different level of processing. For example, parsing or on itself, let's consider pronunciation of the word like discount. If it exists as noun we would emphasize on the *dis* during pronunciations for example DIScount while if it is in the form of verb we would emphasize on *count* in short disCOUNT hence ease the work of text to speech conversion (Jurafsky et al 2006).

Kĩkamba (Kamba) is a Bantu language spoken by almost four million Kamba people in Kenya according to the 2009 Population & Housing Census (Oparanya, 2010). Most of this population lives in the Machakos and Kitui counties and a substantial number along the Embu, Taita Taveta and Tharaka boundaries. For a long time the Kamba people have preserved their culture through carving, especially at Wamuyu and also basketry (kĩondo) not to forget their dance (kĩlumi). The Akamba Culture Trust (ACT) formed in 2005, is crusading for the preservation of culture through written form in their literature and research departments. Despite the efforts of the organization and the number of people speaking the language, Kĩkamba still lacks basic language technology resources and tools. Only recently a spell checker was developed at the School of Computing & Informatics of the University of Nairobi in Kenya. This project focuses on the development of a Named Entity Recognizer for Kĩkamba. Having a good NER system for this language is useful for a wide range of applications, such as event detection with an emphasis on map and phrase browsing, information retrieval and general data mining. Building a successful NER system cannot really be done without an accurate part-of-speech tagger, unfortunately not available for Kĩkamba. In this dissertation we will outline how a part-of-speech tagger and name entity recognizer can be built with a minimum of human effort, using annotated corpora and language independent, state-of-the-art machine learning techniques.

The PoS and NER will be examined in this project using supervised memory based approach which is the automatic machine learning away from the old method of hand crafted rules

1.1 Problem Statement:

Kĩkamba language lacks tools for analysis and synthesis of the language, generations of pre-view, learning (part of speech), event detection and information retrieval for its increasing online textual genre and digital libraries. The Akamba Culture Trust (ACT) is crusading for more Kĩkamba Literature and preservation of the culture inform of written format and of late there is a lot of interest in our mother tongue language in kenya. Hence, to be able to effectively pass the information to the next generation and the interested parties, it is my concern to develop this tool to make the work easier for future generation.

The key issues of Kĩkamba PoS and NER that are of key importance for this project that we want to examine are:- identify tag classes namely: noun, pronoun, adjective, verb, adverb, preposition , punctuation, interjection, conjunction , exclamation and name entity class which is organized in form of locations, personal names and organization from a document(s).

An investigation of Kĩkamba language will be made to find out which characteristics of the language can make construction of the classifier easier or hard and what the limitation of the whole process are?

1.1.0 Objectives:

Overall objective

- Develop a Kĩkamba language tool that will help to analyse and provide categorical pre-views of the language increasing soft textual genre.

Specific objective

- Investigate the Kĩkamba language part of speech tagging and their behavior and come up with tagger which can predict future tags for new words from unstructured documents (textual genre) database.

- To deliver name entity classifier for Kĩkamba language based on IOB standards tags which will be used to classify future proper names from unstructured database(several documents) textual genre.

- Investigate the morphology (behaviour, function and structure) of Kĩkamba language and identify the elements which make it easy and hard to deliver the above mentioned objectives.

1.2 Project Justification

- The growth of digital libraries and the Internet in size and complexity, poses users with greater need to get a sense of the scope and contents of information resources (unstructured text). This project is supposed to ease by providing way to extract information and be able to grasp the overall contents of documents and collections of various papers. The literature department of Akamba Culture Trust (ACT) is crusading the need for growth of textual genre, but with the years of neglect of the language, then language tools are needed if a wider fraternity is to benefit from this initiative, and also primary schools which teach mother tongue, hence this project.

- Since Bantu languages are related, it will provide a frame work which can be extended to other bantu language or the whole classifier architecture can be modified to be a multi-lingual classifier

- As it will be pointed in the language review, work on NER for many of African language is yet to be out of researchers hand while for PoS tagging only a few have been published, therefore this challenge becomes one of my motivation to develop a Kĩkamba classifier so as to contribute in the pool of knowledge for natural language.

2.0 LITERATURE REVIEW

2.1 Introduction

In the current digital era, the extraction and classification of large volumes of Information from text present in multiple unstructured documents e.g. journals, newspapers; magazine etc, has become pertinent issue due to the increased need for access to knowledge resources in format easy to summarize. The Web evolution particularly in areas like Social Networking (face book, twitter, you tube and linked in) has generated large volumes of information from which knowledge needs to be extracted and utilized. How to organize the information to accessible knowledge is a key area of application for Natural Language Processing (NLP). Query of database (structured knowledge base and data mining techniques could not be able to solve this problem without further modification. When the above challenge was posed during Sixth Message Understanding Conference (MUC-6) (R. Grishman & Sundheim, 1996), it bore the field of name entity recognition which utilizes part of speech tagging. The emphasis was on extraction of personal name, place location and organization which make the proper names.

By definition part of speech tagging is assigning each word in a sentence or corpus to it most appropriate morpho-syntatic category from the one listed in the lexicon of the language in question (noun, verb etc for English). Part of speech tagging is also known as word classes, morphological classes, or lexical tags. For it to done, the corpus need to be tokenized so that punctuations are separated from words, what is known as disambiguating end of a sentence. The tags help when you want to process words which belong to a certain class in a special way and also in a superficial disambiguation process which is crucial in parsing. (Zarvel et al, 1999)

Named entity recognition (NER), also known as Name entity extraction (NE), Name entity Name entity (NE) detection, NE tagging or NE identification, is to recognize structured information, such as proper names (person, location and organization), time (date and time) and Numerical values (currency and percentage) from natural language text (Fei Huang , 2005)

2.2 Survey of Languages

A wide variety of languages have been examined in the Context of named entity recognition (Nadeau and Sekine, 2007) and part-of-speech tagging, but very few sub-Saharan African languages have such tools available to them. Part-of-speech Tagging has been investigated in the South African language context. A number of tag sets and preliminary systems are available for Setswana (Van Rooy and Pretorius,2003), Xhosa (Allwood et al., 2003), Northern Sotho (Prinsloo and Heid, 2005; Taljard and Bosch,2005; de Schryver and De Pauw, 2007; Faaß, 2010). Outside of South Africa, POS tagging for Swahili has been extensively researched using finite-state techniques (Hurskainen, 2004) and machine learning methods (De Pauw et al., 2006) and some preliminary experiments on Luo have been described in De Pauw et al. (2010).Swahili is also - to the best of our knowledge -the only Bantu language that has been studied in the context of named-entity recognition (Shah et al., 2010). A few research efforts however investigate the problem of recognizing African named entities in regional varieties of English, such as South African English (newspaper articles) (Louis et al.,2006) and Ugandan English (legal texts) (Kitoogo et al., 2008).

2.3 Application of Part of Speech

- In a document or multiples of them PoS tagging gives large amount of information about the word and the possible neighbor (n-gram and bi-gram). In English, for example, if a word lexical tags are a possessive pronoun e.g. ***My***, it is likely to be followed by noun. For example, my box is stolen. ***My*** is a possessive pronoun while ***box*** is a noun. This kind of phenomena is of great importance if we are to have language model (conceptual model) for speech recognition.
- In speech synthesis and accurate speech recognition systems, word class is used to indicate the correct pronunciations of words e.g. word like "discount" if it exists as a noun in a sentence, we would put emphasize on the **dis** in

pronunciation for example DIScount while if it is in the form of verb we would put emphasize on **count** in short, disCOUNT. (Jurafsky et al 2006).

- PoS tagging are of great importance when it comes to information retrieval since most of name entity proper names are noun in nature. Therefore you can zero in to the subset of nouns from the universal word classes possible thereby reducing greatly the resources (memory, bandwidth and time complexity) that will be used to return the needed output.

2.4 Application of NER

NER as a wide application especially due to the fact that it deals with unstructured data/information which for long time there was no automatic way of gathering information from them. It continues to excite reseachers and as days go by we expect more application. However, in the meantime the current application are namely: event detection, information retrival and data mining.

2.4.1 Event detection.

Documents provide wealth of information in unstructured way e.g Suppose we have a collection of conference papers and you want to know something about the conference like the location where the conference took place(I-LOC), conference name(I-ORG) and the person who presented a certain paper(I-PER) then since collection of journal or the conference papers would form sources of large heterogenous information resource (textual genre) from which we need to analyse, the name entity would be the best to address this challenge. Event detection are not only applicable in conference papers but also in news papers, collection of e-books, company papers, goverment gazzete, but also in various strategic plans. Smith presented a paper (corpus was collection of war and battles text) whose content mainly involved map and phrase browing whereby in map browsing with an interface you could select a range of period(time), then a map would be drawn of location(I-LOC) where battle and war happening in the window frame would be selected while in phrase browsing still using the interface and window frame it gives you key phrases. By clicking it you are able to learn more about it. E.g Post election chaos in 2007/2008 in kenya. (Smith, 2002)

Swan extend the aspect of event detection to the analysis of key entities in corpus over a time period. For example, in Kenya, if you take a collection of daily news papers between 2007 and 2009 then one of the key elements would be post election chaos. Then by applying Swan idea we would have President Kibaki and Raila compete for election, results announced by electroral comissioner Kivuitu, chaos erupts all over Kenya, Annan former United nation chief jets in to mediate peace, Raila and kibaki form goverment, Prosecutor Moreno comes in to discuss way for justice ..etc (Swan et al ,1999) this idea is know as *time varying entities analysis.*

2.4.2 Information retrieval

One branch of information retrieval is question answering as argued by Srihari.(Srihari & Li, 1999). According to them the key to question processor is understanding the question asking point which is usually (*who, what, when, where,* etc). By looking at the sentence *WHO* did *WHAT* (to *WHOM*) *WHEN* and *WHERE*, If we apply name entity classification WHO and WHOM would be the equivalent of I-PER or I-ORG, while WHERE is I-LOC . A study conducted by Srihari for the TREC-8 competition showed that out of the 200 questions asked 80% were asked for name entity response. Here are some rules which can be used to guide the question answering session.

Who/whom --> PERSON
When --> TIME/DATE
Where/what place --> LOCATION
What time (of day) --> TIME
What day (of the week) --> DAY
What/which month --> MONTH
What age/how old --> AGE
What brand --> PRODUCT
What --> NAME
How far/tall/high --> LENGTH
How large/big/small --> AREA
How heavy --> WEIGHT
How rich --> MONEY
How often --> FREQUENCY
How many --> NUMBER

How long --> LENGTH/DURATION
Why/for what --> REASON

On the other hand we have semantic information retrieval whereby a Boolean conventional query is taken has input and a list of web pages or related elements is produced. For example, if we take a query like statistical packages we would have a return of "SAS," "SPSS," "Minitab," "BMDP" and "R" (Nadeau, 2007). Application on this segment was done on biomedical data as presented by (Leser et al, 2005) The NER was to solve three different problems: The reorganization of the entities within the text in a document, assignment of class for each entity (gene, protein, drug etc) and the selection of the preferred term for naming the object in case that a synonyms exist. The major entities in this project consisted of gene or protein name, diseases, drugs, mutation or properties of protein structure. The entities were annotated and then used to develop the biomedical NER which then was used to analyze a large collection of information e.g. Medline database which contained 15 million abstracts and was growing at a rate of 400 000 articles per year. However in the construction of the biomedical NER, they experienced challenges such as lack of general naming convention and frequent use of abbreviation and uses of synonyms.

2.4.3 Data mining

Extraction of information from the web pages poses challenges because of the unstructured definition, their un-trusted sources and their dynamic changing nature. Name entity is used to build search methodology (web/text mining) based on the *redundancy of information* that characterizes the Web, allowing us to detect important concepts and relationships for a domain through a statistical analysis. Moreover, the exhaustive use of web search engines represents a great help for selecting "representative" resources and getting global statistics of concepts; they can be considered as our particular "experts" in all domains (Sanchez & Moreno, 2005).

2.5 Types of Learning for the PoS and NER classifier:

2.5.1 Supervised learning:

Based on the concept of learning with a teacher, whereby a deduction of a function (f) is made from training examples (d) in which the training examples consist of inputs (X) and desired outputs(Y). The values of function f is gotten from a certain fraction of training data **d** by finding hypothesis h from members of **d** which agrees with **f**, in which the hypothesis **h** will be best guess of f. There is also consideration of the inductive biases in arriving at hypothesis **h**. This approach is more of instructive learning, the teacher or critic has the knowledge about the environment in which the learner will be trained on. The teacher is dismissed once the learner is well trained. The approaches in this class include hidden Markov models, memory based learning etc

2.5.2 Unsupervised learning.

Unsupervised learning is a deep concept that can be approached from very different perspectives, from psychology and cognitive science to engineering. It is sometimes referred to as "learning without a teacher". This implies that a learning human, animal, or artificial system observe their surroundings and, based on these observations, adapts their behavior without being told how to associate given observations to given desired responses (supervised learning) or without even being given any hints about the goodness of a given response (reinforcement learning). Usually, the result of unsupervised learning is a new explanation or representation of the observation data, which will then lead to improved future responses or decisions (Kitoogo, 2009 Phd Thesis).For the purpose of name entity recognition the techniques lies on lexical resources such as word net, on lexical patterns and statistical computed on large annotated corpus.

2.5.3 Semi-supervised learning

The technique is also called weak learning which is a recent method which involves some minimal level of supervision then it can generalize thereafter. The method in this category is bootstrapping. A set of seed is required to initiate the learning process. Let's take a scenario where the interest is to learn Kĩkamba Animal names. The seeds will be a few names of animals say twenty animals for that matter which the system will request. After that minimal supervision then the system will try to find cases of animal's names that appear in related context. Then the newly found instance they used to find names and certain generalization are made and the whole process is repeated. At the end of the day a whole knowledge base for animals is established.

2.6 Approach for Building PoS and NER

2.6.1 Rule based/grammar based:

It works in two stages mainly. Use of a dictionary to assign each word a list of potential part of speech then uses a large list of hand written disambiguation rules to winnow down this list to a single part of speech for each word. In stage two, it mainly requires hand crafted rules and a lot of human work and skill. Though it leads to high accuracy it has a setback of being not portable.

2.6.2 HMM (hidden Markov model)

This is a special case of Bayesian inference where the observations are sequences of words, the part of speech tag are the classes which we are supposed to assign to the observation and heavily depended on probabilities. It builds a bi-gram language model for each next name category (Predict next name category from previous word and its name category) and uses Viterbi search to find class tag assignment to corpus with highest probability.

2.6.3 Maximum entropy model (MEM)

When you have various information sources then MEM becomes the most suitable. MEM allows the computation of probabilty of a given feature given hypothesis $p(f|h)$ for any feature f in the space of possible futures, F, and for every hypothesis h in the space of possible histories, H. Futures are defined as the possible classification and a history is all of the conditioning data which enables us to make a decision in the space of futures. The computation of $p(f|h)$ is dependent on a set of features which are binary functions of the histroy and future.

2.6.4 Memory based Learning.

This is a supervised lazy learning algorithm for classification also called similarity based, example based, case based, instance based excellent for evaluating real valued or discrete function. The algorithms works by storing the training labels in the memory then when an instance query is encountered related, set of training labels are retrieved and used to classify this new query. One of it is variant k nearest neighbor (k-nn) which is of key importance to us because of its incarnation in Tilburg memory based learner (TiMBL) toolkit. The memory based approach was chosen because the other method (ruled based, hidden Markov model and Maximum entropy) suffer from sparse data effect. The words which are in test data and not in training data are given probability zero. Thus to cross over this deep valley in the journey of the construction of the tagger we have to use a smoothing strategy to estimate the probability of the unknowns words/events which is present in TiMBL and the other approaches limits on the types, amounts and information that they can take into account in that features of the models are represented as state hence when more rich text format is used it leads to explosion of the state. But MBL which is wrapper of Timbl software solves the above problems by use of implicit similarity based smoothing scheme and rich feature set by automatic weighting

2.7 Architecture of The Kīkamba PoS and NER

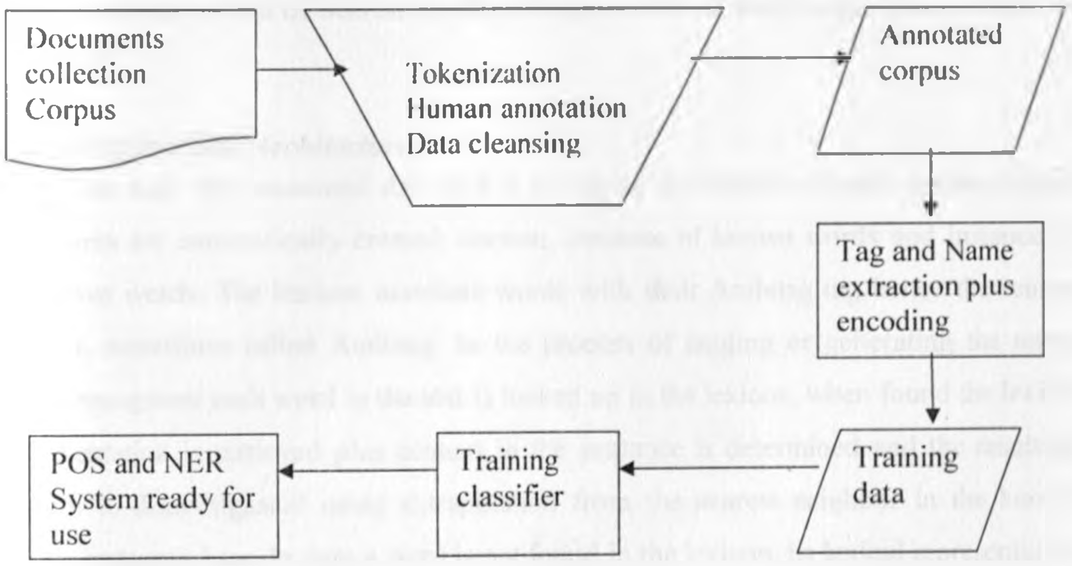

Documents collection Corpus
Tokenization
Human annotation
Data cleansing
Annotated corpus
Tag and Name extraction plus encoding
Training data
Training classifier
POS and NER System ready for use

Fig 2.1 Architecture for the classifiers

2.8 Algorithms

2.8.0 Software

Sun java virtual box is a collection of powerful virtual machine tool. The virtualization uses both software and hardware virtualization. The machine has host operating system which in our case will be widow vista or Window XP. While our guest operating system will be Linux (ubuntu).The virtual box comes with already installed TIMBL and its two wrappers Memory based tagger (Mbt) and Memory based tagger generator (Mbtg). Therefore, after installing the guest operating system you go directly to terminal and run the command invoking Mbtg and Mbt. Sun java virtualization tool which can be found at (http://www.virtualbox.org).

A Microsoft application excels will be used in the task of manual annotation of unique identifiers of entities from the corpus. Then annotated corpus will be transferred to a text file (notepad) will be used in the window environment while Linux text file will be used when running the NER. In Timbl the algorithms used is memory based whereby it stores the training set explicitly and classification of the testing cases is done by extrapolation from the most similar cases this is the key hypothesis of the algorithm. The behavior of MBL is similar to that of nearest neighbor hence a child of K-NN algorithms. (Cover et al, 1967).

2.8.1 Mbtg and Mbt Architecture

Once we have the annotated data and it is run in the memory based systems, three structures are automatically created: lexicon, instance of known words and instance of unknown words. The lexicon associate words with their Ambitag tag hence the reason lexicon sometimes called Ambitag. In the process of tagging or generating the name entity recognizer each word in the text is looked up in the lexicon, when found the lexical representation is retrieved plus context in the sentence is determined and the resulting pattern is disambiguated using extrapolation from the nearest neighbor in the known words instances base. In case a word is not found in the lexicon, its lexical representation is computed on the basis of its form, its context is determined and the resulting pattern is disambiguated using extrapolation from the nearest neighbor in the unknown instance base .The output in the above scenario is always the best guess for the word in the current context.

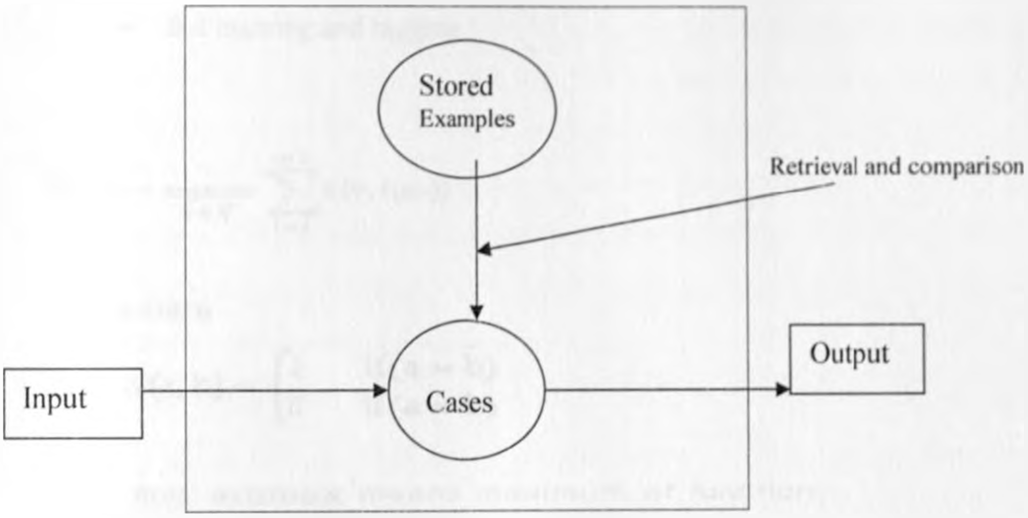


Fig 2.2 Working of memory based systems

2.8.2 k- nearest neighbor learning.

A problem in this category is stated as given a set of N- points in D-dimensional space and unlabelled examples xn≡ R^n then we find the point which minimizes distance x_n The Euclidean distance is used to calculate this minimal distance between the new instance to be classified and the set of similar training instances. For example let instance x be described by feature vector

$< a_1(x), a_2(x), a_3(x), \ldots\ldots\ldots\ldots\ldots a_n(x)>$

Where *ar(x)* denotes the value of the rth attribute of instance *x*. Then the distance between two instances x_i and x_j is defined to be $d(x_i,x_j)$, where

$$d(x_i,x_j) \equiv \sqrt{\sum_{r=1}^{r=n} \left[a_r(x_i) - a_r(x_j)\right]^2}$$

This method harbors the following advantages;

- a relatively small data set can be sufficient for training,
- incremental learning,
- explanation capabilities,
- its non-parametric nature, and

- fast learning and tagging

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{i=k} \delta(v, f(x_i))$$

where

$$\delta(a, b) = \begin{cases} 1 & \text{if}(a = b) \\ 0 & \text{if}(a \neq b) \end{cases}$$

and ***argmax*** means maximum of function

The MBL Memory based learner systems consist of two components namely learning and the performance component. The learning components just add training instances to the memory thus the reason is referred as lazy learner. An instance consists of a fixed-length vector of n feature-value pairs, and an information field containing the classification of that particular feature-value vector. (Walter et al, 2007) while the performance components classification is done by mapping inputs to outputs. Let's say we have a new instance B while all set of examples in memory are A. Then to calculate the similarity between instances A and B we need distances metrics $\Delta(A,B)$. Extrapolation is done by assigning the most frequent subset category of the found set of similar examples. Breaking resolution method is used to resolve tie categories which we will discuss in section2.9.4.

2.8.3 TREE

In this project we chose memory based learning approaches whereby instances are stored in form of array where we need to search from beginning to end in case we meet a query. Our toolkit TIMBL while implementing its two packages MBT and MBTG incorporate trees as the mode of storing the instance base so as to save classification time and storage space. Various types of tree will be discussed below

- **IBI** It represents the set of training instances in the tree. To order the tree, it divides the information gain by the number of values. This means you start with the feature with the highest gain ratio. Incase information gain is shared by two features look for the one which has different number of values. An exhaustively search is done which uses k current distance encountered to heuristically optimize the search. In this tree instances are stored as the path while the feature values as the arcs. Classification is the node which the arcs lead to.

- **IB2 incremental editing** Only the instances that play positive roles are kept in memory while others which play no roles are disruptive for classification and are discarded or edited from memory.
- **IGTREE** The instance based is compressed into a tree structure whereby information gain is used for the compression (determine the order of the feature). It uses top down search with no backtracking and prunes arcs which leads to the same classification.
- **TRIBL** Tribl is a hybrid of IB1 and IGTREE. This works by starting as Igtree for the first x features with the highest information gain as supplied in the experiment in question, on x+1th feature acts as IB1 tree. Feature weight is used to build the tree.
- **TRIBL2** This assumes all behaviors of TRIBL tree. However it switches to IB1 only when a mismatch is encountered by Igtree.

2.9 Metrics Used In Memory Based Systems.

2.9.1 Overlap metrics

Δ (A, B) is distances between the instances A and B which is represented by n features and δ the distance per feature. In short from the equation below it is the sum of the differences of the feature. According to Aha any K-nn algorithms, such metrics is referred as IB1(Aha et al, 1991).However in our case here the k represent the k-nearest distance rather than k-nearest examples as the case was with the original version.

$$\Delta(A,B)=\sum_{i=1}^{n}\delta(a_i,b_i)$$

$$\delta(a_i,b_i) = \begin{cases} abs\frac{a_i b_i}{max_i\ min_i}\ ifnumeric,else \\ 0ifx_i = y_i \\ 1ifx_i \neq y_i \end{cases}$$

The short coming of this metrics is either all or nothing as indicated in the equation. Therefore for strings which may be similar in eyes of the expert, the algorithms will output it as a mismatch for example goats and goat because of 's'. Since this metric counts the number of (Mis) matches of the numerical features values in both patterns absence of them poses threat to the metric.

2.9.2 Information-gain (IG) and gain ratio feature weighting

IG is statistical property, which measure how well a given attribute separate the training examples according to target classification. To define it completely we use another component called entropy which characterizes the Im(purity) of an arbitrary collection of examples (Mitchell, 1997).

$$\omega i = H(C) - \sum_{v \in V_i} P(v) \times H(C \mid v)$$

Whereby C is the set of class labels. V_i set of values for feature i and H(C) = −Pc2C P(c) log2 P(c) is the entropy of the class labels. From the training set relative frequencies the probabilities are estimated. However to avoid overestimation of the of relevance features with large numbers value, we introduce split info which normalize the information gain resulting to gain ration as per the equation below.

$$\Delta(X,Y) = \sum_{i=1}^{n} \omega i \delta(xi, yi)$$

Where the w_i is the weight metrics

The gain ratio can be used to calculate distance as per the equations above resulting to K-nn algorithm called IB1-IG (Daelemans et al, 1992) the ability to estimate the probability (relevance) of features implies that many different features can be added to the features set.

2.9.3 CHI square

The best known goodness of fit test is referred as chi- square and is calculated as follows. It introduces no bias and can be compared across conditions in numbers degree of freedom.

$$\chi 2 = \sum_{i} \sum_{j} \frac{(Eij - Oij)2}{Eij}$$

Where Oij is the observed number of cases with value vi in class cj, i.e., $O_{ij} = n_{ij}$, and E_{ij} is the expected number of cases which should be in cell (v_i, c_j) in the contingency table

2.9.4 Tie breaking.

In K-NN classifier, when choosing the majority category ties can occur frequently i.e. two or more of majority class has the same number of features. In order to resolve this tie in TIMBL, first, the value of the k parameter is incremented by 1, and the additional nearest neighbors at this new K^th distance are added to the current nearest neighbor set (k is subsequently reset to its user-specified value). If the tie in the class distribution persists, then the class label is selected with the highest overall occurrence in the training set. If that is also equal, then the first class is taken that was encountered when reading the training instance file. (Walter et al, 2007)

2.10 Annotation

The work on the Kĩkamba tagger was annotated manually the speech tags in excel sheet with the format as shown below. The tags namely were noun, verb, adverb, adjective, preposition, punctuation, interjection and conjunction.

Table 2.1 Example of annotation

WORD	POS	NER
Usumbĩ	Noun	I-ORG
wa	preposition	O
Ngai	Noun	I-PER
Nĩ	Conjuction	O
Kyaũ	Adjective	O
?	Punc	O
I	N/A	O

On the other hand NER annotation will use the Ramshaw and Marcus (1995) tagging scheme of IOB where (O) indicates a word is outside of an entity , (I) inside an entity and (B) at the beginning of an entity. It has major interest in three entities namely Persons (PER), Organizations (ORG), Locations (LOC).

2.10.1 Person Names

- First, middle and last names of people
- Titles such as "Mr." and role names such as "President" are **NOT** considered part of a person name.
- Appositives such as "Jr.", "Sr.", and "III" *are* considered part of a person name.

2.10.2 Location Names

- Roads (streets, motorways)
- Regions (villages, towns, cities, provinces, countries, continents, dioceses, parishes)
- Structures (bridges, ports, dams)

- Natural locations (mountains, mountain ranges, woods, rivers, wells, fields, valleys, gardens, nature reserves, allotments, beaches, national parks)
- Public places (squares, opera houses, museums, schools, markets, airports, stations, swimming pools, hospitals, sports facilities, youth centers, parks, town halls, theaters, cinemas, galleries, camping grounds, NASA launch pads, club houses, universities, libraries, churches, medical centers, parking lots, playgrounds, cemeteries)
- Commercial places (chemists, pubs, restaurants, depots, hostels, hotels, industrial parks, nightclubs, music venues)
- Assorted buildings (houses, monasteries, crèches, mills, army barracks, castles, retirement homes, towers, halls, rooms, vicarages, courtyards)

2.10.3 Organization Names

- Companies (press agencies, studios, banks, stock markets, manufacturers, cooperatives) Sub-divisions of companies (newsrooms)
- Political Organizations (political parties, terrorist organizations)
- Government bodies (ministries, councils, courts, political unions of countries (e.g. U.N., E.A.C., A.U., etc.)
- Publications (magazines, newspapers, journals)
- Musical companies (bands, choirs, opera companies, orchestras
- Public organizations (schools, universities, charities)
- Other collections of people (sports clubs, sports teams, associations, theaters companies, religious orders, youth organizations)

2.11 Evaluations Metrics

In classifying process, if we assume that we have class C, the following subclass occurs: The TP or true positives cell contains a count of examples that have class C and are predicted to have this class correctly by the classifier. The FP or false positives cell contains a count of examples of a different class that the classifier incorrectly classified as C. The FN or false negatives cell contains examples of class C for which the classifier predicted a different class label than C. (Walter et al 1997) and TN true negatives cell contains a count of examples that are not of class C but the classifier predicted them to be of this class C .From the above the true positives examples are given by P=TP + FN while the inverse is F=FP+TN. Then the following criteria of performance can be calculated.

2.11.1 Precision:

Is the number of correct named entities divided by the number of named entities found by the learning system (percentage of named entities found by the learning system that are correct) or the proportional number of times that the classifier as correctly made decision some instances are in class C.

$$\boxed{precison(P) = \frac{TP}{TP + FP}}$$

2.11.2 Recall:

This is the number of correct named entities divided by the total number of named entities in the data [corpus] (percentage of named entities present in the corpus that are found by the system) or the proportional number of times the classifier assign class C of test data instances, also referred to as true positive rate (tpr).

$$\mathrm{Re}call = \frac{TP}{P}$$

2.11.3 Harmonic mean F-score

This is weighted harmonic mean of recall and precision

F-score= (2×precision×recall) / (precision + recall).

2.12.0 Validation

Since our ojective is to create a predictive classifier and estimate how accurate the predictive model will perform in actual enviroment and noting the scarceity of the corpus, then employment of K-fold cross validation methodoloy (Weiss and kulikowski, 1991) will be done. This will allow the use of the available data to find the best configuration for the classifier.This methodology is motivated by model selection(learning parameters (optimal),weights and the numbers of neighbour in k-NN) and performance estimation (what is the true error rate of the entire system). In our case the data is divided into ten fold, were 9-fold will be used as training data and the 1-fold as the test data

3.0 TAGGERS DEVELOPMENT

3.1 Data

3.1.0 Introduction.

Kĩkamba is one subset of Bantu, mostly spoken in the lower part of Eastern Province, the regions namely Machakos, Kitui, Mwingi and Makueni . It has variety of dialects, in our case, since available corpus is religious data, Masaku dialect which is the major dialect and the one used to write the Kikamba bible is the one in use in this tagger. Most of the data was collected from the manuscripts of Jehovah Witness since there is scarcity of Kĩkamba digital online materials. However, from the emphasis for digital data by the Akamba Trust Culture (ACT), soon a lot of it will be available and further exploration of this language will be done easily.

3.1.2 Data preparation

The data collected was not 100% pure hence some residue had to be removed so as to work on a clean filtrate which would have a positive impact on accuracy of the overall tagger developed for this project. Some of the impurities included wrong punctuation, wrongly inserted words, corrupted information but not limited to these. The corpus was in prose and had to be changed to a format that Memory based tagger (MBT) could understand (format of the tagger generation input (one word per line, two columns for word and corresponding tag) can be used Mbt_3.1_manual.pdf page 10). In order to meet the above mentioned, employment of Microsoft excel was used to help in formatting one word per cell in one column, sentence after sentence up to the end. This is called tokenization whereby words are separated from punctuation to ensure each word is alone. Two excel were prepared for the part of speech and another for the name entity recognizer respectively. The actual manual annotation was done during the excel stage. In total the Kĩkamba corpus has a total of 27754 words and a part of it, almost 2000 words is a seed of nouns made to boot trap the name entity recognizer since the corpus was not well endowed for the NER. Table 3.1 is an example of the format together with the annotation part:

Table 3.1 Extract from excel of annotated PoS and NER.

part of speech

INDEX	WORD	POS
1	Usumbi	noun
2	wa	preposition
3	Ngai	noun
4	Ni	preposition
5	Kyau	pronoun
6	?	punc
7	<utt>	
8	Ngusi	noun
9	sya	adjective
10	Yeova	noun

Name entity recognizer

INDEX	WORD	NER
1	Usumbi	O
2	wa	O
3	Ngai	I-PER
4	Ni	O
5	Kyau	O
6	?	O
7	<utt>	
8	Ngusi	O
9	sya	O

3.2 Annotation

Each part of speech tagging (PoS) and name entity recognizer (NER) had different class which the corpus was to be annotated to. Namely for PoS: adjective, noun preposition, verb, pronoun, Interjection, num, adverb, punc and conjunction. Some of them have been abbreviated for example punc supposed to represent punctuations and num is supposed to be the short form of numbers. On the other side of coin NER had the following classes:O,I-LOC,I-PER,I-ORG,B-PER, num, B-LOC for further explanation on them one by one and fully guidelines used in the annotation of this taggers kindly consider section 2.10 of this manuscript. The annotation was done in Microsoft excel in operating system window Vista, but the actual system runs in Linux (Made using ubuntu10.0) though can run in any version of Linux so long as Timbl, Timbl server, MBTG and MBT are installed. Hence the annotated corpus transferred to Linux put in a text file for PoS tagger was named testgen.txt and for Name entity recognizer was named nergen.txt.

3.3 Stages in formation of the taggers

The figure 3.1 shows the steps the annotated corpus goes through before actual tagger can be run. Annotated corpus is put in, a frequency lexicon tag file is created which contains the words with different possible tags it has been assigned in the corpus plus the frequency of occurrence of each in the corpus, resulting to what is referred as Ambitag (An ambitag is a symbolic label defining for a word the different tags it can have according to the corpus, Mbt_3_1_Manual.pdf page 6). Then, a case base for known words is created using known words ,Timbl parameter.The user specified for known words are used at this point which includes tree for storage, metric e.g information gain e.t.c. This known case base is used to classify any known words that are present in the text that the user will input. The unkown words that the user inputs are classifed using case base for unknown words(which the ambitag are not available) which is created immediately after the one for known words, The parameter specified for unknown words are used here. Finally, the information for all those files are stored in a settings file and it is in this file which is used to run in memory based tagger(Mbt) software package.

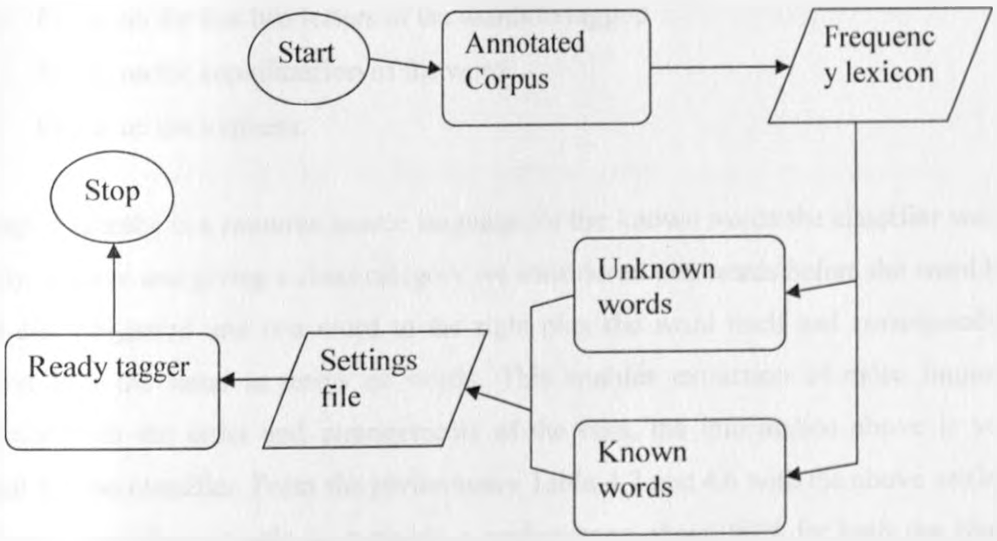


Fig 3.1 the stages MBTG uses to develop taggers

3.4 Form and context

Words (tokens) from the corpus actually belonged to two groups namely known and unknown groups. Several parameters were used to capture the form and the context of the word in focus. This enabled capturing of key elements present in the language which would influences future predictions of different categories and their disambiguation in the classifies. The parameters in questions were:

For knowns words group:

- Focus on two disambiguated words on left.
- Focus on one ambitguation tag on the left plus the word been disimbiguated.
- Focus for each context tags two words on the left for the corresponding word.
- Focus for each context tag one word on the right for the corresponding.

For unknowns words group:

- Focus one disimbiguated tag on the left
- Focus one ambiguation tag to the right
- Focus on the first two letters of the word to be tagged
- Focus on the last two letters of the words to tagged
- Focus on the capitilazation of the word
- Focus on the hyphens.

Though Kĩkamba is a resource scarce language,for the known words the classifier would simply retrieve and giving a class category we considered two words before the word has been disambiguated and one word to the right plus the word itself and corresponding context was the same in terms of words. This enables extraction of more lingustic evidence with the order and arrangements of the tags, the information above is very crucial for the classifier. From the performance Table 4.7 and 4.6 with the above settings the known words were able to maintain a performance above 90% for both the Name entity recognition and part of speech tagging. For the unknown words group, we consider only one word on both sides because we had only 27,754 words in the corpus, therefore considering many words as the case with the knowns words would result in a classification problem since there would be a small pool of words to generalize with resulting in poor performance. Even with the little context and form considered, the part of speech tagging returned an average of 71.93% in overall accuracy.

In view of the morphological structure of Kikamba language, for example many words will start with Mb, Nd, Ng, Ny, Th, Mw, Kw, Ky, Ma, Sy, Nth and Ngw just to mention a few but the major as their first two letters e.g Mbui *–goat*, Ndua - *village*, Ng'ombe - *cow*, Nyambu-*wild animal*, Thooa - *price*, Mwaka - *year*, Kwangolya - place name, Kyeni - *light* and place name, Kyalo - persons name, Maiu – *bananas* and Syombua - persons name e.tc thus the reason why we considered the first two letters which are in the focus word. Most names of places in Kikamba language will end with –Ni e,g Kathiani, Kaviani, Nzaikoni Makueni and Mitaboni hence the considering of the last two characters at the end of the word in the focus again. Most of the nouns start with captial letters and since name entity recognizers focus most of the time on noun, capitilisation was also included in the paramaters, many diacritics are present in Kikamba together with hyphen and words which have hyphens tend to have similar structure at the start or at the middle of the words, for example, Ng'aa, Ng'ala, Ng'anga, Ng'eng'eta, Ng'ombe, Ng'ota etc. All of the above was included so as to capture the entirety of all possible structures which would likely help in producing higher accuracy results.

The Timbl parameters in case of known word since the word on focus demanding classification is already presented in the known data of the classifier. Overlap metrics become very effective and efficient in terms of wall clock and computer memory performance and to ensure faster classification we store the known words in a tree which prunes arcs leading to same classification, searches in top down manner and do compression of the stored values which is IGTREE tree. The compression is done using information gain property where words with higher occurrence in particular language are likely to be placed on top and for unknown words information gain was maintained as the property for arranging words in the tribl2 tree which is a combination of IGTREE and IBI and aims to exploit the trade-off between optimization of search speed (as in IGTREE), and maximal generalization accuracy (Daelemans et al 1997). Modified value difference(MDVM) is the parameter which has been used for unknown words. MDVM is a method to determine the similarity of the values of a feature by looking at co-occurrence of values with the target classes.(Timbl-6_0_3 user guide page 28). The similarity being considered makes it easier to handle uknown words encountered in cause of classification

The variable mentioned above helps to create a classifier which is efficient, effective in terms of computer wall clock and performance hence mode of optimization.

3.5 Actual taggers

The classifiers were developed from MBTG(Memory based tagger generator) and the resulting main files testgen.settings and nergen.settings were the real classifiers for part of speech tagger and name entity recognizer tagger respectively which could used in Mbt(Memory based tagger) for testing new data. All this can be done in the terminal and to cater for user who have a phobia for terminal (command prompt), a graphical user interface (GUI) was developed in python version2.6.

3.6 Graphical user interface (GUI)

The GUI consists of three textboxes one for input and the rest for output. There are two buttons one for submitting the input to the classifiers or taggers. One submitted the output will come automatically each on its own textbox according the MBT classifier. Consider figure 3.2. The title of the GUI is Kamba name entity recognizer and part of speech, just below the title there is instructions on how to input text on the input textbox labeled 1. Below the input text box on the left side is a submission button labeled submit, once you enter the text you submit it to the MBT using this button. On the right side there is clearing button labeled clear, once there is text in any textbox or all of there you use this button to clear.

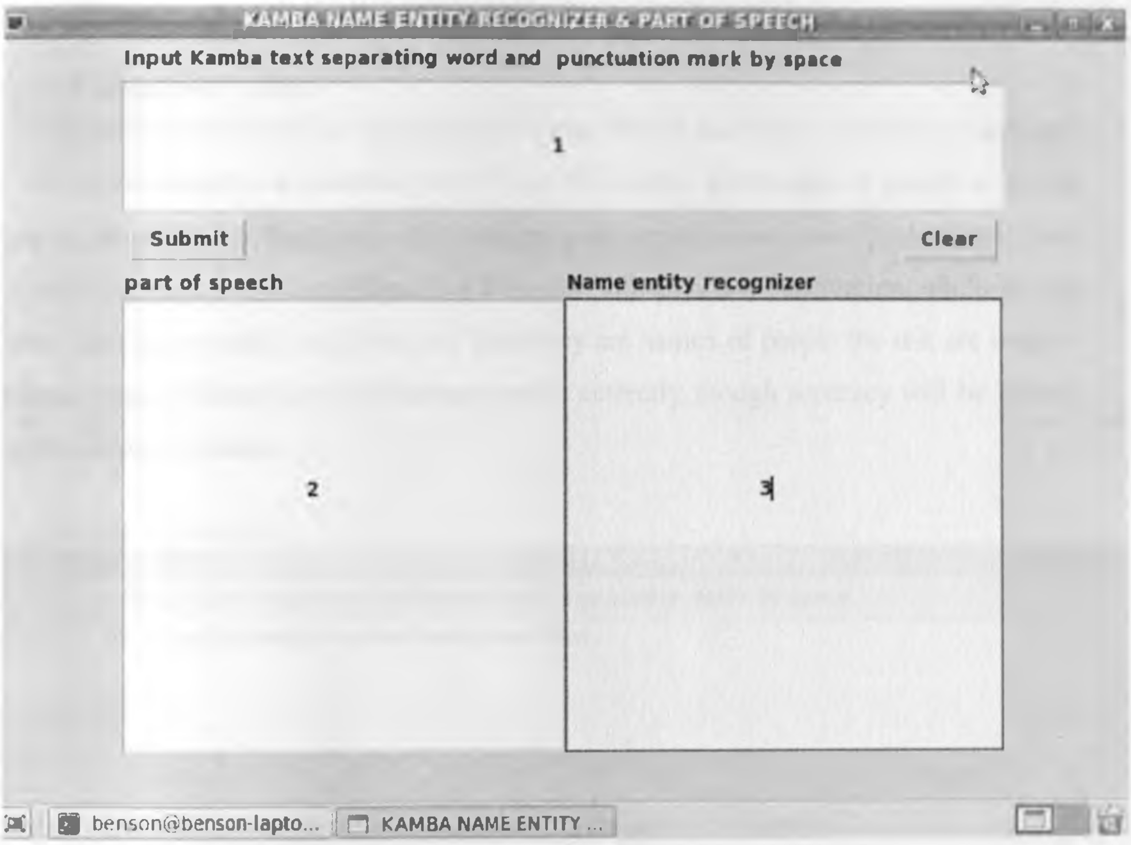

KAMBA NAME ENTITY RECOGNIZER & PART OF SPEECH
Input Kamba text separating word and punctuation mark by space
1
Submit
Clear
part of speech
Name entity recognizer
2
3
benson@benson-lapto...
KAMBA NAME ENTITY ...

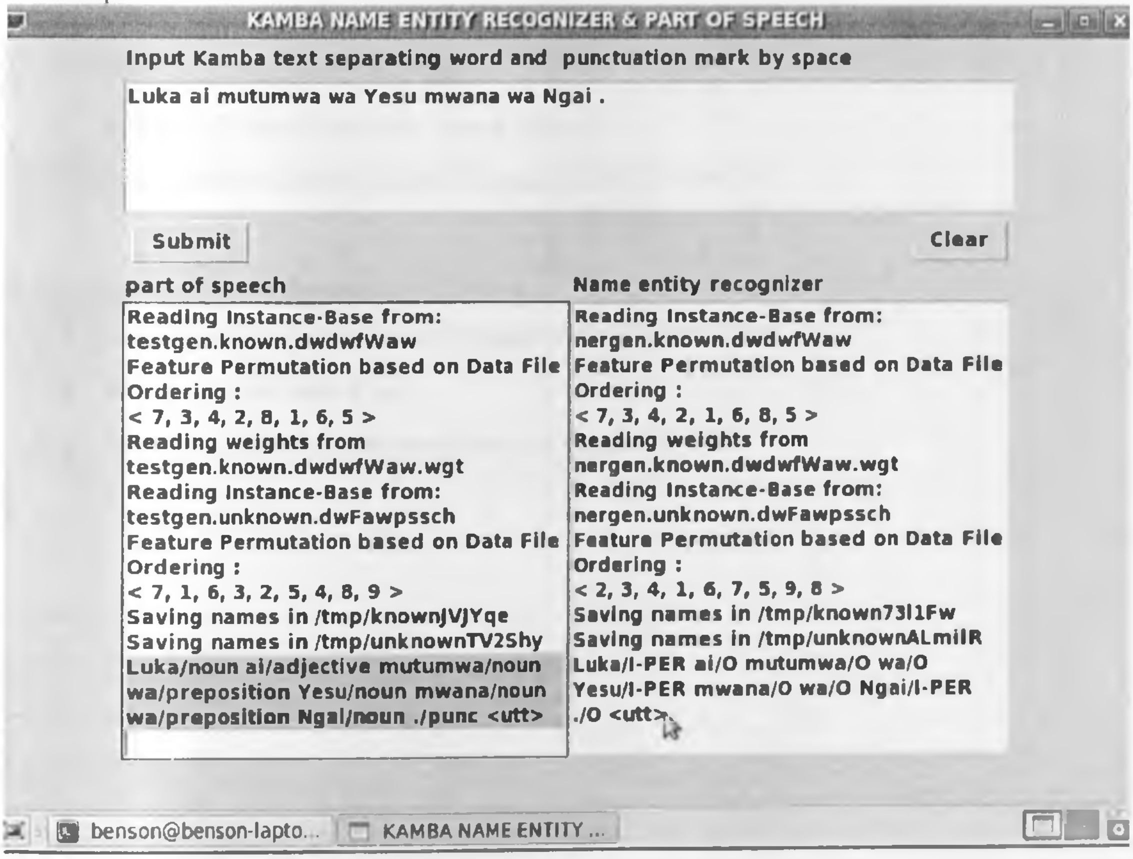
Fig 3.2 Graphical user interface (GUI) of the classifiers.

The text box with number 2 is the one for output for the part of speech while the one on the left side with number 3 is for outputting the name entity recognizer.

3.6.1 Examples of a Run

The inputted text "Luka" ai mutumwa wa Yesu Mwana wa Ngai . "Which is translated Luke was an apostle of Jesus the son of God . The output for the part of speech is luka is a noun, ai which describes Luke is an adjective, mutumwa a noun, wa a preposition, Yesu a noun, mwana a noun and Ngai is a noun the full stop is a punctuation, while on the other side of the coin Luka,Yesu and Ngai they are names of people the rest are outside which is true. Hence the classifier can predict correctly though accuracy will be looked upon during evaluation.

An example of screen shot


KAMBA NAME ENTITY RECOGNIZER & PART OF SPEECH
Input Kamba text separating word and punctuation mark by space
Submit
Clear
part of speech
Name entity recognizer
benson@benson-lapto...
KAMBA NAME ENTITY ...

```
Luka ai mutumwa wa Yesu mwana wa Ngai .
```

```
Reading Instance-Base from:
testgen.known.dwdwfWaw
Feature Permutation based on Data File
Ordering :
< 7, 3, 4, 2, 8, 1, 6, 5 >
Reading weights from
testgen.known.dwdwfWaw.wgt
Reading Instance-Base from:
testgen.unknown.dwFawpssch
Feature Permutation based on Data File
Ordering :
< 7, 1, 6, 3, 2, 5, 4, 8, 9 >
Saving names in /tmp/knownJVJYqe
Saving names in /tmp/unknownTV2Shy
Luka/noun ai/adjective mutumwa/noun
wa/preposition Yesu/noun mwana/noun
wa/preposition Ngai/noun ./punc <utt>
```

```
Reading Instance-Base from:
nergen.known.dwdwfWaw
Feature Permutation based on Data File
Ordering :
< 7, 3, 4, 2, 1, 6, 8, 5 >
Reading weights from
nergen.known.dwdwfWaw.wgt
Reading Instance-Base from:
nergen.unknown.dwFawpssch
Feature Permutation based on Data File
Ordering :
< 2, 3, 4, 1, 6, 7, 5, 9, 8 >
Saving names in /tmp/known73l1Fw
Saving names in /tmp/unknownALmilR
Luka/I-PER ai/O mutumwa/O wa/O
Yesu/I-PER mwana/O wa/O Ngai/I-PER
./O <utt>
```

Fig 3.3 Example of a run of the taggers

3.7 Requirement and How to Run the Tagger:

The following specification and above will do well for this tagger:

- Pentium Central processing unit(C.P.U) 1ghz and above
- Ram should be 1 GB and above.
- Operating system UNIX or Linux

The following programs are needed:-

- Timbl- 6.3.0 and above
- Timbl server-1.0.0
- Mbt3.1.3(which include Mbtg
- Python 2.6(should include tkinter library)

The first three can be found at http://ilk.uvt.nl/timbl/ for free.

3.7.1 Running the Taggers:

- Copy the folder containing the tagger into your machine,
- Make sure the path is set.
- On the terminal inside the folder run tagger.py.

4.0 EXPERIMENTS RESULTS AND THEIR ANALYSIS

4.1 Methodology

The main aspects were data gathering and tagger development. Data gathering involves understanding of name entity recognition and part of speech tagging, literature review which involves the concepts, feature, algorithms and their evaluation and previous work done on the subject, corpus collections which may be gotten by various methods of data mining (web mining), from books, from publishers and specific people, interviewing Kĩkamba Linguists so as to understand Kĩkamba language more. Tagger development involves cleansing, tokenization and annotation of the corpus, classifier development and finally evaluation.

The *k*-folds model suggested by Weiss (Weiss and Kulikowski, 1991) was used to perform evaluation of the system; the method was selected because of the relatively small size of the Kikamba corpus. We used a *k* value of 10. The annotated corpus is going to be portioned in ten equal folds and ensured it was done at the sentence boundary, so as not to affect the context of a word. The resulting partitioned corpus was partitioned into two parts one containing 9 parts which was 90% of the corpus, which was for the training set and the remaining 10% used as the test set. This process was repeated ten times each time ensuring not to repeat any partition. An average of the 10 runs was generated for the final evaluation.

4.2 Evaluation metrics

Recall, precision, harmonic mean F score (summarize precision and recall in one measure.), true positive, false positive, true negative and false negative which have been explained in section2.11 will be used. In addition extraction of: False positive rate(FPR) which is a ration of false positives and total negatives, AUC area under the curve which is defined by the two dimensions FPR (false positive rate, x axis)and TPR (true positive rate, or recall, y-axis).(Timbl_6_0 manual pgs 33). Then F-scores and AUC scores are micro-averaging and macro-averaging. In micro-averaging, each class' F-score or AUC is weighted proportionally to the frequency of the class in the test set. A macro-average adds all the F-scores or AUCs and divides this sum by the number of classes in the training set, finally, confusion matrix associates the class predicted by tagger (vertically) with the real class of the test items given (horizontally).

4.3 Experiment results

For the purpose our discussion here, data will be presented in summary and details extracted from every run from the first one to the tenth test will be found in section Appendix a

4.3.1 Data for the part of speech.

Table 4.1 Scores per class of the part of speech

Scores per class

class |	precision	Recall(TPR)	FPR	F-score	AUC
noun |	0.77679	0.97072	0.18538	0.86269	0.89267
preposition |	0.94701	0.94282	0.00679	0.94456	0.96802
pronoun |	0.98277	0.85629	0.00027	0.91394	0.92801
punc |	0.99705	0.98282	0.00043	0.98978	0.99119
adjective |	0.95677	0.90407	0.00441	0.92912	0.94983
conjuction |	0.84605	0.93234	0.00585	0.88623	0.96324
verb |	0.86144	0.41123	0.00838	0.55465	0.70142
interjection |	0.85560	0.78332	0.00330	0.81175	0.89001
adverb |	0.94955	0.84857	0.00088	0.89449	0.92385
num |	0.98309	0.32485	0.00028	0.47663	0.66228
exclamation |	0.00000	0.00000	0.00000	0.00000	0.00000
AVERAGE	**83.24%**	**72.34%**	**1.96%**	**75.13%**	**80.64%**

Table 4.2 Average accuracy scores for PoS

Average for each test

	F-Score beta=1		AUC,		ACCURACY
TESTS	Microav	Macroav	Microav	Macroav	overall accuracy:
TEST1	0.82682	0.80417	0.88584	0.84411	92.07%
TEST2	0.84281	0.81413	0.88855	0.85036	90.64%
TEST3	0.84626	0.81475	0.89111	0.85742	90.22%
TEST4	0.84703	0.82203	0.89202	0.86217	90.75%
TEST5	0.84711	0.82506	0.89263	0.87743	89.74%
TEST6	0.84940	0.83034	0.89400	0.88116	90.41%
TEST7	0.85372	0.83116	0.89898	0.88341	90.43%
TEST8	0.85379	0.83665	0.89914	0.88825	91.06%
TEST9	0.85916	0.84088	0.90130	0.88951	92.69%
TEST10	0.86544	0.84467	0.90168	0.89531	88.74%
AVERAGE	**84.92%**	**82.64%**	**89.45%**	**87.29%**	90.68%

4.3.2: Data for name entity recognizer

Table 4.3 Scores per class of NER

Scores per class

class |	precision	Recall(TPR)	FPR	F-score	AUC
O |	0.98323	0.99853	0.13820	0.99081	0.93017
I-LOC |	0.96492	0.84945	0.00118	0.90160	0.92414
I-ORG |	0.93823	0.92857	0.00060	0.93109	0.96399
I-PER |	0.97983	0.86687	0.00134	0.91848	0.93277
B-ORG |	0.92222	0.84865	0.00033	0.87910	0.92416
B-PER |	0.00000	0.00000	0.00107	0.00054	0.00000
B-LOC |	1.00000	0.73571	0.00000	0.82476	0.86786
num |	0.00000	0.00000	0.00005	0.00005	0.00000
AVERAGE	96.47%	87.13%	2.04%	68.08%	92.38%

Table 4.4 Average accuracy scores for PoS

Average for each test

	F-Score beta=1		AUC,		ACCURACY
TESTS	Microav	Macroav	Microav	Macroav	overall accuracy:
TEST1	0.96524	0.79824	0.87534	0.78315	98.07%
TEST2	0.97400	0.88259	0.89988	0.79546	96.21%
TEST3	0.98261	0.89173	0.90800	0.81252	97.56%
TEST4	0.98278	0.89863	0.92998	0.81278	98.07%
TEST5	0.98421	0.93834	0.93209	0.81624	97.48%
TEST6	0.98501	0.94760	0.93543	0.81907	98.58%
TEST7	0.98581	0.94922	0.94229	0.82126	98.64%
TEST8	0.98768	0.94940	0.94240	0.85401	98.31%
TEST9	0.98959	0.95591	0.95088	0.88070	96.75%
TEST10	0.97516	0.95974	0.98052	0.88980	98.42%
AVERAGE	98.12%	91.71%	92.97%	82.85%	97.88%

4.4 Analysis

The Table 4.1 and 4.3 indicate a precision of 83.24% and 96.47% for part of speech tagging and Name entity recognizer respectively, for the part of speech tagging there is indication there was substitial false positive classifiation which lowered the percentage and a close look at individual class category for part of speech in the confusion matrix extracted from MBT indicated that noun and preposition were the classes which had alot of false positives with the noun class leading. However on the name entity recognizer the least false negatives were seen, which were contributed by class 'O" with the other classes for both classifier doing very well. The recipricoal of total positives (true positive and false negatives) mutiplied true positive result to the measure recall and as we have stated in discussion for precision that noun, preposition and class "O" the mis-classified automatically it quives the mis-categorisation results into false negatives hence the reason the classifier returns a 72.34% and 87.13% for Part of speech tagging and Name entity recognizer respectively.The mis-classification for former seems to too many because of the low percentages. Finaly, From the same Table 4.1 and 4.3 we have F-score which is a weigheted harmonic score for the recall and precision for the classifiers which stands at 82.64% and 91.71% for part of speech and name entity recognizer classifier. This experimental results are encouraging. Kikamba and Kiswahili are both Bantu languages. The two languages are closely related and share morphology. Kiswahili F-score is 81.5% for name entity recognizer classifiers (Shah.R et al 2010) as compared to 91.71% for Kikamba here.

4.4.1. Nouns analysis

Nouns have done poorly by recording the least percentage of 77.68% in terms of precision from the table 4.1. Indeed from the confusion matrix available in Appendix a noun was Mis-classified as other classes 202 times out of 875 noun in test one, 227 times out 962 available nouns in test two, 199 out 926 in the third run, 167 out 918 in 4th run,159 out of 907 in the 5th run,199 out of 913 in the 6th run, 224 out of 919 nouns in the 7th , 204 out 937 in the 8th run while the 9th run had 254 out of 994 and finally 231 out of 898. Mostly the nouns were mis-classified to class of numbers and verbs and this can be seen clearly in the confusion matrix. The above has been captured in chart below.

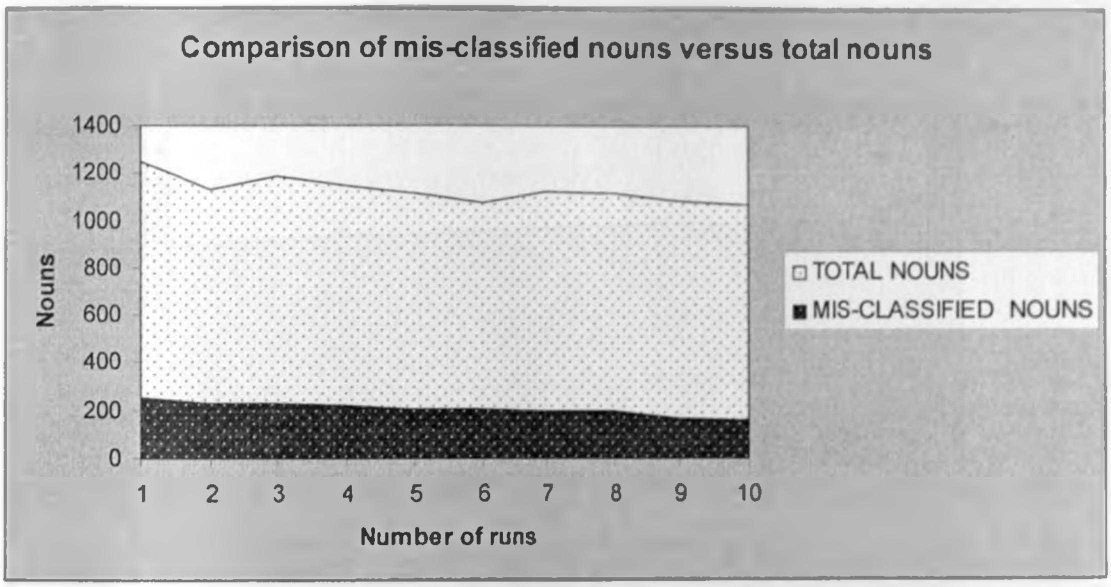


Fig 4.1 Mis-classified nouns versus total nouns

Table 4.5 Numbers of words used versus overall accuracy.

numbers of words	accuracy overall
2k	77.77
4k	79.26
6k	81.15
8k	81.98
10k	83.64
12k	84.53
14k	85.25
16k	85.58
18k	86.14
20k	96.01

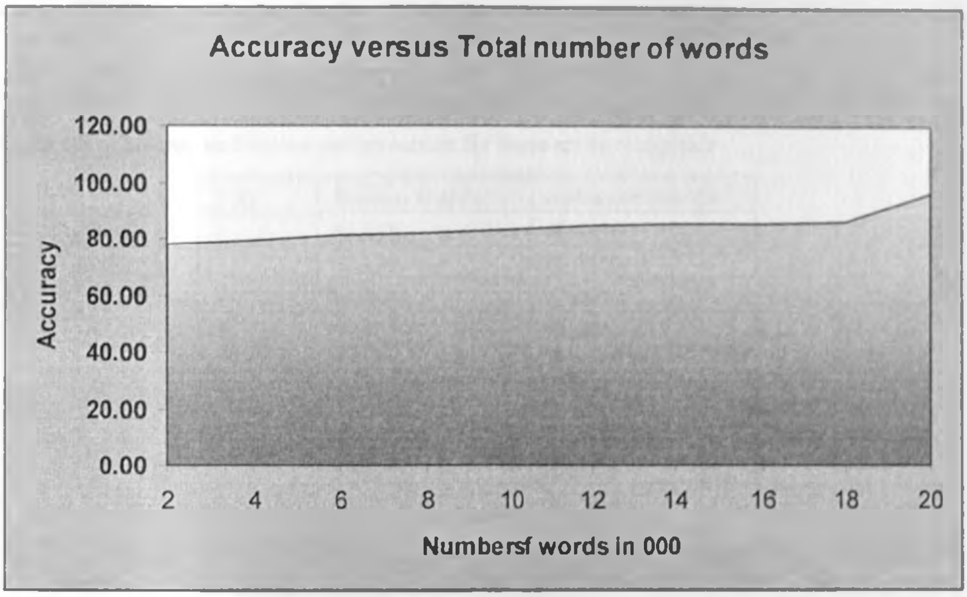


Fig 4.2 Accuracy versus total numbers of words

From figure 4.2 the line indicates that accuracy is directly proportional to corpus available. The larger in terms of corpus words, the better the result of evaluation. This experiment was done with corpus starting with 2000 words with increment of 2000 words up to 20000 words, The indented purpose was to show that since the overall accuracy of the Pos tagger was 90.68% , if More corpus is gotten and added over to the tagger , its accuracy is bound increase

Table 4.6 known and unknown words performance for part of speech tagging

Test	Known words	Unknown words
1	94.24%	78.01%
2	94.59%	73.65%
3	94.55%	71.31%
4	94.79%	68.44%
5	93.44%	71.43%
6	94.34%	68.62%
7	95.85%	67.05%
8	95.46%	70.25%
9	95.42%	83.64%
10	93.80%	66.86%
Average	**94.65%**	**71.93%**

Table 4.7 unknown and known performances for name entity recognizer

Test	Known words	Unknown words
1	98.81%	98.44%
2	98.13%	88.07%
3	98.60%	93.00%
4	98.67%	94.68%
5	98.77%	91.28%
6	98.76%	97.56%
7	99.33%	95.73%
8	98.81%	95.96%
9	98.59%	90.63%
10	99.13%	95.39%
Average	98.76%	94.07%

4.5 Performance of unknown and known words of the tagger

Looking at the overall performance of the classifier and memory based tagger generated as its dual side in generating classifier namely the known and unknown words. A closer look at the results for the part of speech tagging in Table 4.6 we note 94.65% and 71.93% as the accuracies for known and unknown respectively. The higher percentages for known words is a clear indication of good retrieval of the classifier. For unknown words the 70% plus performance is a encouraging keeping in mind that Kikamba is a resource scare language with many diacritics. However, to aid in good generalization of the categories for the class we included approximately 2,000 words as a bootstrap seed for both classifier. We hope future classification of unseen text will improve greatly. Comparing the Swahili part of speech tagging which gave a result of 98.46% and 91.61% for known and unknown words (De Pauw et al, 2006) and overall performance of 98.25% compared with ours which has recorded 90.68%. The performance on Swahili very high compared withour performace on Kikamba. There appears to be a close correlation between the size of the corpus and the accuracy. The Swahili corpus size was 3,656,821 words in comparison with our Kikamba corpus of size 27,754 words.

On the name entity recognizer, which was our key classifier we report a performance of 98.76% and 94.07% for known and unknown words as from table 4.7. This is good performance for a Bantu language name entity recognizer and opens the window for more research in Bantu related language because it is apparent that with a small cropus, we can still achieve good results using the memory based approach. The overall performance was 97.88%, a very good result.

4.6 Limitation

In principle and in practice there is usually a difference and that is why mathematician would need a standard deviation measure to see how much and find out why. In course of our tool development life cycle, some obstacles were encountered which reduced our velocity of tagger development and accuracy. Some of them were resolved while others were not. These obstacles are:

- *Scarcity of the* Kĩkamba *corpus.* Kĩkamba being a language limited to three counties and only used mostly for communication purpose, there is little written of it, mostly available is the bible. The online available Kĩkamba language stuff is very few and again mostly available is religious data hence out of 27754 words used in the developed of the taggers 90% is religious data meaning the resultant taggers are more inclined to religious side and not wholesome of Kĩkamba literature. The corpus gotten had a lot of repetition hence reducing the number of words available.

- The data had a lot of mistakes especially the one gotten through web mining; hence applying soap and water to cleanse it was not an easy task. Most of the people in town are not fluent speakers and writers of Kĩkamba and even in kamba land only a few old folks who can locate mistakes easily and getting one was like chasing a loose goose in the woods. With few people who really understand the language, a Kĩkamba dictionary came handy to increase the speed of cleansing

- *The tool lacks a spell checker.* When a user interacts with the system and by chance makes a typing error, the system (taggers) will not recognize it and will move to classify despite the error resulting to classification error.

- Kĩkamba *has variety of dialects. The* most acceptable one for writing was Machakos which implies that it is the only dialect which has been taken care of since all available corpus was in that dialect. However, most other dialects only differ by only about 5%. Thus the tool is still useful to them.

5.0 CONCLUSION AND RECOMMENDATION

Knowledge and information has increased exponential in cause of time in contrary the human beings have relied more on systems than before. Indeed, everyday man looks forward to the day "with just a click of button on a remote control, the daily chores will be achieved automatically." The state described above has forced the computing world especially the artificial intelligence researcher to look and tailor solution that can fill the wide gap stated. Large portions of monetary have been allocated to such research programs and with hope a lot is expected in future that will make communication easier and faster in the global village. In future it seems it will be more of machine learning than human learning.

In this dissertation, exploration of Kĩkamba as a Natural language using machine learning methods (very instrumental in artificial learning) approach was pipelined and the resultant effect was two classifiers, namely Part of Speech tagger (PoS) and Name entity recognizer tagger (NER). The TIMBL toolkit as described along the thesis was the key in the implementation and provided the necessary ingredients for this project, Now that the table is set to munch the taggers and in future someone can use and modify to suit other needs as may arise. In the evening of this project the stated below becomes the lessons and entry point of this project to the world of Natural languages processing linguistic research.

5.1. Contribution

The umbrella contribution of this dissertation was to demonstrate, explore and implement use of machine learning (supervised and boot trap learning) to develop a Natural language processing tool (name entity tagger and part of speech tagger) for Kĩkamba language. Specific contributions are hereby stated.

- A gap was identified in literature review of only few languages in Africa having been researcher on and most of them risk being lost because of the global village effects where people are only learning and using national languages of their respective countries. Kĩkamba having these two classifiers becomes preserved for future generation: The tool will provide the optimum curiosity needed to excite one to examine Kĩkamba language more, with help of ACT Akamba Culture Trust organization campaign the research will result into generation of more tools and extra online Kĩkamba data then the language will be preserved for future generation.

- The tool enables events detection in various Kĩkamba manuscript using the name entity recognizer likewise, the same tagger can be used in information retrieval based on the organization, location of places and people names,

- Modified further it will aid in web mining of data which is a key component in information retrieval: the tool can be incorporated into building automatic information retrieval and mining systems (the tool can form part of search engine for software like Google if a Kĩkamba one was to be implemented). The Part of speech tagger can aid in building speech synthesis and speech recognizes system, which is of great significance in computer and human interaction and communication (this is a key objective of artificial intelligence where a computer can think and act like human being "the Turing machine"). Hence if extended to Kĩkamba is a major stride.

- The tool becomes Kĩkamba teaching aid: The government is in gestation period of creating a digital village. The ICT Park is being build at Malili in Kamba land around Machakos. For schools having computers this will become an added advantage because the tool will aid in teaching. For example pupils and teachers of lower elementary level where Kĩkamba language is taught up to standard three can utilize the tool to understand part of speech and possible names related entities. For those who teach adult class in Kĩkamba language and non Kĩkamba speaking people who want to understand the language can use the tool to achieve the same objective as the category above. Finally the students and researchers in linguistics will find the tool useful either to consolidate or explore new dimension whether in Kĩkamba language or another language in course of their research or study.

- The paper 'B. Kituku, P. Wagacha, G. Pauw 2011, *In proceeding of Human language technology development conference*, A Memory based approach to Kĩkamba Name entity recognition pg 106-111,Alexandrina, Egypt' is product of this research.

- Finally, the tool would help to market Akamba people in tourisms within and without Kenya.

5.2 Conclusion

We have presented the Name entity recognition tagger with accuracy of 97.88% and harmonic F-score of 91.71% together with embedded part of speech tagging with overall accuracy of 90. 68% and harmonic F-score of 82.64%. For the part of speech tagging it becomes the second Bantu classifier after the Swahili one so far. While for the Name entity recognizer classifier becomes the first Data driven tagger for Bantu and second in terms general Name entity recognizer classifier after the Swahili online translator one.Now with this classifier available, we have a suitable language resource for use by linguists, researchers and system developers. As stated earlier, the part of speech tagging classifier compared with the Swahili one, the latter performed better therefore some of the future work will include increasing the size of the corpus and investigating the relationship between the size of the corpus and the classifier accuracy. We can extend lessons learned here to other Bantu languages. Apart from Bantu languages in the East African region, we have Nilotes and Cushites languages which to the best of our knowledge have no classifiers for POS tagging and Name entity recognition. Future work will look into these languages.

5.3 Recommendation and future work.

The child "Kĩkamba Name entity recognition tagger and Part of speech tagger" tool has been born, however along the gestation period some aspect with side effects were identified and need to fixed by the next researcher and there some addition modification which can be made to the tool to suite various needs as per user. Hence to the natural language processing and Machine language researcher, the following recommendations are made as future work:

For the accuracy of the user to be improved at confidential level, a spell checker is needed to ensure that once the user puts a wrong spelled words immediately s/he is prompted to rectify before processing hence such a plug in is of much needed. It can be made in such a manner it can fit in other language not only Kĩkamba.

In order to fully make use of the tool as a teaching aid, addition of morphological analyzer is needed so that words are carefully examined and the learner can understand how to generate a word form the root – morphemes and how probabilities of letters following each other.

There are more languages in Kenya which need tools to be developed; Hope researchers will be able to invest a lot on Natural language processing and develop tools for other languages more so, using different approaches so that later comparison of the best approach for African language can be gotten.

References

Allwood,J. Grönqvist, L and Hendrikse, A ,P 2003. Developing a tagset and tagger for the African languages of South Africa with special reference to Xhosa. Southern African Linguistics and Applied Language Studies, 21(4):223–237.

Aha, D. W, Kibler, D, & Albert, M 1991. Instance-based learning algorithms.Machine Learning, 6, 37–66.

Borthwick, Andrew, Sterling, J. Agichtein, E. and Grishman, R 1998. NYU: Description of the MENE Named Entity System as used in MUC-7. *Proc. Seventh Message Understanding Conference.*

Chinchor, Nancy 1999. Overview of MUC-7/MET-2. *Proc. Message Understanding Conference MUC-7.*

Chinchor, Nancy, Robinson, P and Brown, E 1998. Hub-4 Named Entity Task Definition. *Proc. DARPA Broadcast News Workshop.*

Cover, T. M, & Hart, P. E 1967. Nearest neighbor pattern classification. Institute of Electrical and Electronics Engineers Transactions on Information Theory, 13, 21–27.

Daelemans, W, Zavrel, J, Berck, S, and Gillis, S. 1996. Mbt: A memory-based part of speech tagger-generator. In Ejerhed, E. and Dagan, I., editors, *Proceedings of the Fourth Workshop on Very Large Corpora*, pages 14.27.

Daelemans, W. Zavrel, J.van der Sloot, V and van den Bosch, 2002)."MBT: Memory- Based Tagger, version 1.0, Reference Guide," ILK Technical Report ILK-0209, University of Tilburg, The Netherlands. 2002.

Daelemans, W. Zavrel, J. Berck,P. and Gillis,S 1996."MBT: A memory-based part of speech tagger generator," in Proceedings of the Fourth Workshop on Very Large Corpora, E. Ejerhed and I. Dagan, Eds., 1996

Daniel. Jurafsky & James H. Martin, 2006. Speech and Language Processing: An introduction to natural language processing,computational linguistics, and speech recognition. Copyright 2006,

Daelemans, W, Zavrel, J. Van den Bosch, A. and Van der Sloot, K. 2007. **Reference Guide** (15 pages, 100 kB PDF); BT: Memory-Based Tagger, version 3.1, Reference Guide. ILK Technical Report Series 07-08.

Daelemans, Walter, Jakub Zavrel, Ko van der Sloot and Antal van den Bosch, 1999. "TiMBL: Tilburg Memory Based Learner, version 4.0, Reference Guide", ILK Technical Report 01-04.

De Pauw,G and De Schryver, G.M and Wagacha,P,W 2006. ("Data-Driven Part-of-Speech Tagging of Kiswahili," *in Proceedings of Text, Speech and Dialogue, 9th International Conference*, vol 4188, Lecture Notes in Computer Science,P. Sojka, I. Kopecek, K. Pala, Eds. Verlag:Springer, 2006.

Faaß,G. 2010. The verbal phrase of northern sotho: A morpho-syntactic perspective. In G. De Pauw, H.J. Groenewald, and G-M. de Schryver, editors, *Proceedings of the Second Workshop on African Language Technology (AfLaT 2010)*, pages 37–42, Valletta, Malta. European Language Resources Association (ELRA).

Grishman R and Sundheim,B 1996. Message understanding conference-6: a brief history. *In Proceedings of the 16th conference on Computational linguistics* - Volume 1, COLING '96, pages 466–471, Stroudsburg, PA, USA. Association for Computational Linguistics

Jakub Zavrel and Walter Daelemans,1999."Recent Advances in Memory-Based Part-of-Speech Tagging." in: Actas del VI Simposio Internacional de Comunicacion Social, Santiago de Cuba, pp. 590-597, 1999, ILK pub: ILK-9903.

Hurskainen, A 2004. HCS 2004 – Helsinki Corpus of Swahili Compilers: Institute for Asian and African Studies (University of Helsinki) and CSC

Louis, A.& de Waal, A & Venter, C 2006, Named Entity Recognition in a South African Context Proceedings of SAICSIT 2006, Pages 170 –179.

Nadeau, D and Sekine, S 2007. A Survey of named entity recognition and classification. Lingvisticae Investigationes, 30(1):3–26, January

Oparanya,W,A . 2010. 2009 Population & Housing Census Results. Available from [http://www.knbs.or.ke/Census Results/Presentation by Minister for Planning revised.pdf], Nairobi, Kenya.

Prinsloo,J and Heid, U 2005. Creating word class tagged corpora for Northern Sotho by linguistically informed bootstrapping. *In Proceedings of the Conference on Lesser Used Languages & Computer Linguistics (LULCL-2005)*, Bozen/Bolzano, Italy

Ramshaw, L.A. & Marcus, M.P., (1995) "Text Chunking using Transformation-Based Learning", ACL Third Workshop on Very Large Corpora, pp. 82-94, 1995

Rau, Lisa F. 1991. Extracting Company Names from Text. *Proc. Conference on Artificial Intelligence Applications of IEEE.*

Srihari, Rohini and Li, W. 1999. Information Extraction Supported Question Answering. *Proc. Text Retrieval Conference.*

Sánchez, David and Moreno, A. 2005. Web Mining Techniques for Automatic Discovery of Medical Knowledge. *Proc. Conference on Artificial Intelligence in Medicine*

Swan, Russell and Allan, J. 1999. Extracting Significant Time Varying Features from Text. *Proc. International Conference on Information Knowledge Management*

Shah, R. , Bo Lin, Anatole Gershman, and Frederking Robert. SYNERGY: A Named Entity Recognition System for Resource-scarce Languages such as Swahili using Online Machine Translation. Proceedings of the Second Workshop on African Language Technology (AfLaT 2010)

T. Mitchell, 1997. *Machine Learning*. New York: McGraw-Hill, 1997.

Tjong,E,J Kim Sang, and De Meulder, F 2003 "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition," *in Proceedings of International Conference On Computational Linguistics* (CoNLL 2003),2003.

Taljard,E and Bosch,S.E 2005. A comparison of approaches towards word class tagging: disjunctively vs conjunctively written Bantu languages. *In Proceedings of the Conference on Lesser Used Languages & Computer Linguistics* (LULCL-2005), Bozen/Bolzano, Italy.

Tjong Kim Sang, Erik. F. 2002. Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition. *Proc. Conference on Natural Language Learning.*

Wang, Liang-Jyh; Li, W.-C. and Chang, C.-H. 1992. Recognizing Unregistered Names for Mandarin Word Identification. *Proc. International Conference on Computational Linguistics.*

Weiss, S., & Kulikowski, C. 1991. *Computer systems that learn.* San Mateo, CA:Morgan Kaufmann.

Zavrel, J. and Daelemans, W. 1997. Memory-based learning: Using similarity for smoothing.In Cohen, P. R. and Wahlster, W., editors, *Proceedings of the Thirty-Fifth Annual Meeting of them Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, pages 436.443, Somerset, New Jersey. Association for Computational Linguistics.

Appendix A

Testing statitics for the Kikamba PoS *and* NER.

PART ONE: PART OF SPEECH EVALUATION DATA.

1. ***Result of testing with training file train1p and test file tpos1***

Scores per Value Class

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
noun |	673	202	920	9	0.76914	0.98680	0.18004	0.86448	0.90338
preposition |	187	13	1594	10	0.93500	0.94924	0.00809	0.94207	0.97057
punc |	252	0	1551	1	1.00000	0.99605	0.00000	0.99802	0.99802
verb |	95	16	1571	122	0.85586	0.43779	0.01008	0.57927	0.71385
adverb |	31	1	1766	6	0.96875	0.83784	0.00057	0.89855	0.91864
num |	29	0	1729	46	1.00000	0.38667	0.00000	0.55769	0.69333
interjection |	43	4	1739	18	0.91489	0.70492	0.00229	0.79630	0.85131
adjective |	158	4	1610	32	0.97531	0.83158	0.00248	0.89773	0.91455
conjuction |	41	10	1749	4	0.80392	0.91111	0.00569	0.85417	0.95271
pronoun |	45	0	1757	2	1.00000	0.95745	0.00000	0.97826	0.97872
exclamation|	0	0	1804	0	(nan)		(nan)		0.00000

Average for the train

```
F-Score beta=1, microav: 0.853716
F-Score beta=1, macroav: 0.836653
AUC, microav:            0.899141
AUC, macroav:            0.889510
overall accuracy:        0.920700  (1554/1804), of which 1563 exact
matches
There were 12 ties of which 5 (41.67%) were correctly resolved
```

Confusion Matrix:

```
          noun preposition   punc   verb adverb    num interjection adjective conjuction pronoun exclamation
        --------------------------------------------------------------------------------------
   noun |    673       0       0      6      0      0      0      2      1      0      0
preposition | 5      187       0      2      0      0      1      1      1      0      0
   punc |      0       1     252      0      0      0      0      0      0      0      0
   verb |    120       0       0     95      1      0      0      0      1      0      0
 adverb |      2       1       0      1     31      0      1      1      0      0      0
    num |     45       0       0      0      0     29      1      0      0      0      0
interjection |2        6       0      3      0      0     43      0      7      0      0
adjective |  25        3       0      3      0      0      1    158      0      0      0
conjuction |  2        1       0      1      0      0      0      0     41      0      0
pronoun |      1       1       0      0      0      0      0      0      0     45      0
exclamation | 0        0       0      0      0      0      0      0      0      0      0
[illegible] |  0       0       0      0      0      0      0      0      0      0      0
```

2. ***Result of testing with training file train2p and test file tpos2***

Scores per Value Class:

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
noun |	735	227	889	19	0.76403	0.97480	0.20341	0.85664	0.88570
preposition |	210	10	1636	14	0.95455	0.93750	0.00608	0.94595	0.96571
pronoun |	19	2	1845	4	0.90476	0.82609	0.00108	0.86364	0.91250
punc |	220	2	1647	1	0.99099	0.99548	0.00121	0.99323	0.99713
adjective |	165	4	1681	20	0.97633	0.89189	0.00237	0.93220	0.94476
conjuction |	61	14	1793	2	0.81333	0.96825	0.00775	0.88406	0.98025
verb |	100	8	1634	128	0.92593	0.43860	0.00487	0.59524	0.71686
interjection |	18	4	1839	9	0.81818	0.66667	0.00217	0.73469	0.83225
adverb |	36	1	1821	12	0.97297	0.75000	0.00055	0.84706	0.87473
num |	32	2	1771	65	0.94118	0.32990	0.00113	0.48855	0.66438
exclamation	0	0	1870	0	(nan)		(nan)		0.00000

Average for the train
F-Score beta=1, microav: 0.849396
F-Score beta=1, macroav: 0.814126
AUC, microav: 0.892018
AUC, macroav: 0.877428
overall accuracy: 0.906400 (1596/1870), of which 1517 exact matches
There were 14 ties of which 5 (35.71%) were correctly resolved

Confusion Matrix:

```
                noun preposition pronoun  punc adjective conjuction  verb interjection adverb   num exclamation
            ------------------------------------------------------------------------------------------------
   noun |      735       1          0       0       1         9        7       0          0       1       0
preposition | 8        210          0       1       2         0        0       3          0       0       0
pronoun |        3       1         19       0       0         0        0       0          0       0       0
   punc |        0       1          0     220       0         0        0       0          0       0       0
adjective |     17       0          0       0     165         0        1       1          1       0       0
conjuction |     1       0          0       0       1        61        0       0          0       0       0
   verb |      121       4          2       0       0         0      100       0          0       1       0
interjection |3          1          0       0       0         5        0      18          0       0       0
adverb |        10       2          0       0       0         0        0       0         36       0       0
    num |       64       0          0       1       0         0        0       0          0      32       0
exclamation | 0          0          0       0       0         0        0       0          0       0       0
    -*- |        0       0          0       0       0         0        0       0          0       0       0
```

3. ***Result of testing with training file train3p and test file tpos3***

Scores per Value Class:

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
noun |	727	199	930	36	0.78510	0.95282	0.17626	0.86086	0.88828
preposition	217	8	1642	25	0.96444	0.89669	0.00485	0.92934	0.94592
pronoun |	23	0	1866	3	1.00000	0.88462	0.00000	0.93878	0.94231
punc |	210	1	1679	2	0.99526	0.99057	0.00060	0.99291	0.99499
adjective |	212	7	1658	15	0.96804	0.93392	0.00420	0.95067	0.96486
conjuction |	53	14	1823	2	0.79104	0.96364	0.00762	0.86885	0.97801
verb |	87	20	1663	122	0.81308	0.41627	0.01188	0.55063	0.70219
interjection	45	18	1828	1	0.71429	0.97826	0.00975	0.82569	0.98426
adverb |	31	0	1855	6	1.00000	0.83784	0.00000	0.91176	0.91892
num |	20	0	1817	55	1.00000	0.26667	0.00000	0.42105	0.63333
exclamation	0	0	1892	0	(nan)		(nan)		0.00000

Average for the train
F-Score beta=1, microav: 0.847031

F-Score beta=1, macroav: 0.825055
AUC, microav: 0.893997
AUC, macroav: 0.895306
overall accuracy: 0.902200 (1625/1892), of which 1540 exact matches
There were 17 ties of which 8 (47.06%) were correctly resolved

Confusion Matrix:

```
                noun preposition pronoun  punc adjective conjuction  verb interjection adverb    num exclamation
        ----------------------------------------------------------------------------------------------------
   noun |    727       3        0        0        3        13       17        0        0        0        0
preposition | 6      217        0        0        0         0        1       18        0        0        0
pronoun |      2       1       23        0        0         0        0        0        0        0        0
   punc |      2       0        0      210        0         0        0        0        0        0        0
adjective |   14       1        0        0      212         0        0        0        0        0        0
conjuction |   1       0        0        0        0        53        1        0        0        0        0
   verb |    120       0        0        1        0         1       87        0        0        0        0
interjection |1        0        0        0        0         0        0       45        0        0        0
adverb |       3       1        0        0        2         0        0        0       31        0        0
    num |     50       2        0        0        2         0        1        0        0       20        0
exclamation | 0        0        0        0        0         0        0        0        0        0        0
    -*- |      0       0        0        0        0         0        0        0        0        0        0
```

4. *Result of testing with training file train4p and test file tpos4*

Scores per Value Class

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
noun |	751	167	891	29	0.81808	0.96282	0.15784	0.88457	0.90249
preposition	194	12	1615	17	0.94175	0.91943	0.00738	0.93046	0.95603
pronoun |	39	0	1789	10	1.00000	0.79592	0.00000	0.88636	0.89796
punc |	225	0	1610	3	1.00000	0.98684	0.00000	0.99338	0.99342
adjective |	196	7	1617	18	0.96552	0.91589	0.00431	0.94005	0.95579
conjuction |	52	8	1773	5	0.86667	0.91228	0.00449	0.88889	0.95389
verb |	86	22	1626	104	0.79630	0.45263	0.01335	0.57718	0.71964
interjection	16	4	1813	5	0.80000	0.76190	0.00220	0.78049	0.87985
adverb |	33	0	1799	6	1.00000	0.84615	0.00000	0.91667	0.92308
num |	24	2	1788	24	0.92308	0.50000	0.00112	0.64865	0.74944
exclamation	0	0	1837	1	(nan)		0.00000	0.00000	(nan)

Average for the train

F-Score beta=1, microav: 0.865444
F-Score beta=1, macroav: 0.844669
AUC, microav: 0.901675
AUC, macroav: 0.857417
overall accuracy: 0.907500 (1616/1838), of which 1556 exact matches
There were 10 ties of which 4 (40.00%) were correctly resolved

Confusion Matrix:

```
                noun preposition pronoun  punc adjective conjuction  verb interjection adverb    num exclamation
        ----------------------------------------------------------------------------------------------------
   noun |    751       1        0        0        2         7       17        0        0        2        0
```

```
preposition |13     194      0      0      0      0      0      4      0      0      0
pronoun |      4      2     39      0      2      0      2      0      0      0      0
  punc |       2      1      0    225      0      0      0      0      0      0      0
adjective |   17      0      0      0    196      0      1      0      0      0      0
conjuction |   3      2      0      0      0     52      0      0      0      0      0
  verb |     101      1      0      0      1      1     86      0      0      0      0
interjection |1       3      0      0      1      0      0     16      0      0      0
adverb |       2      1      0      0      1      0      2      0     33      0      0
   num |      24      0      0      0      0      0      0      0      0     24      0
exclamation | 0       1      0      0      0      0      0      0      0      0      0
   -*- |       0      0      0      0      0      0      0      0      0      0      0
```

5. ***Result of testing with training file train5p and test file tpos5***

Scores per Value Class.

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
noun |	748	159	904	21	0.82470	0.97269	0.14958	0.89260	0.91156
preposition	177	25	1619	11	0.87624	0.94149	0.01521	0.90769	0.96314
pronoun |	30	0	1794	8	1.00000	0.78947	0.00000	0.88235	0.89474
punc |	237	0	1591	4	1.00000	0.98340	0.00000	0.99163	0.99170
adjective |	166	5	1640	21	0.97076	0.88770	0.00304	0.92737	0.94233
conjuction |	77	10	1743	2	0.88506	0.97468	0.00570	0.92771	0.98449
verb |	77	20	1637	98	0.79381	0.44000	0.01207	0.56618	0.71396
interjection	32	7	1773	20	0.82051	0.61538	0.00393	0.70330	0.80573
adverb |	28	4	1792	8	0.87500	0.77778	0.00223	0.82353	0.88778
num |	29	1	1764	38	0.96667	0.43284	0.00057	0.59794	0.71613
exclamation	0	0	1832	0	(nan)		(nan)		0.00000

Average for the train
F-Score beta=1, microav: 0.859162
F-Score beta=1, macroav: 0.822030
AUC, microav: 0.901296
AUC, macroav: 0.881156
overall accuracy: 0.89740 (1601/1832), of which 1524 exact matches
There were 17 ties of which 6 (35.29%) were correctly resolved

Confusion Matrix:

```
            noun   preposition pronoun  punc adjective conjuction  verb interjection adverb  num exclamation
          ---------------------------------------------------------------------------------------------------
   noun |     748      0      0      0      0      7     13      0      1      0      0
preposition | 2      177      0      0      2      0      0      6      1      0      0
pronoun |       2      3     30      0      1      0      2      0      0      0      0
  punc |        0      3      0    237      0      0      0      1      0      0      0
adjective |    17      0      0      0    166      0      4      0      0      0      0
conjuction |    1      1      0      0      0     77      0      0      0      0      0
  verb |       94      2      0      0      1      0     77      0      0      1      0
interjection |1       14      0      0      0      3      0     32      2      0      0
adverb |        4      2      0      0      1      0      1      0     28      0      0
   num |       38      0      0      0      0      0      0      0      0     29      0
exclamation | 0        0      0      0      0      0      0      0      0      0      0
   -*- |        0      0      0      0      0      0      0      0      0      0      0
```

6. ***Result of testing with training file train6p and test file tpos6***

Scores per Value Class:

class	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
noun		714	199	959	25	0.78204	0.96617	0.17185	0.86441	0.89716
preposition	243	2	1638	14	0.99184	0.94553	0.00122	0.96813	0.97215
pronoun		40	1	1853	3	0.97561	0.93023	0.00054	0.95238	0.96485
punc		227	0	1668	2	1.00000	0.99127	0.00000	0.99561	0.99563
adjective		179	16	1686	16	0.91795	0.91795	0.00940	0.91795	0.95427
conjuction		62	15	1816	4	0.80519	0.93939	0.00819	0.86713	0.96560
verb		91	12	1651	143	0.88350	0.38889	0.00722	0.54006	0.69084
interjection	33	6	1854	4	0.84615	0.89189	0.00323	0.86842	0.94433
adverb		40	0	1850	7	1.00000	0.85106	0.00000	0.91954	0.92553
num		17	0	1848	32	1.00000	0.34694	0.00000	0.51515	0.67347
exclamation	0	0	1896	1	(nan)		0.00000	0.00000	(nan)

Average for the train
F-Score beta=1, microav: 0.853792
F-Score beta=1, macroav: 0.840878
AUC, microav: 0.898983
AUC, macroav: 0.862167
overall accuracy: 0.90410 (1646/1897), of which 1607 exact matches
There were 17 ties of which 7 (41.18%) were correctly resolved

Confusion Matrix:

```
                noun preposition pronoun   punc adjective conjuction   verb interjection adverb   num exclamation
            ------------------------------------------------------------------------------------------------------
   noun |      714       0         0        0       2         12        11        0         0        0        0
preposition | 6        243         0        0       3          0         0        5         0        0        0
pronoun |        3       0        40        0       0          0         0        0         0        0        0
   punc |        1       0         0      227       0          0         0        1         0        0        0
adjective |     15       0         0        0     179          0         1        0         0        0        0
conjuction |     2       1         0        0       1         62         0        0         0        0        0
   verb |      132       1         0        0       9          1        91        0         0        0        0
interjection |2          0         0        0       0          2         0       33         0        0        0
adverb |         6       0         0        0       1          0         0        0        40        0        0
    num |       32       0         0        0       0          0         0        0         0       17        0
exclamation | 0          0         1        0       0          0         0        0         0        0        0
    -*- |        0       0         0        0       0          0         0        0         0        0        0
```

7. *Result of testing with training file train7p and test file tpos7*

Scores per Value Class:

class		TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC

noun |	695	224	906	15	0 75626	0 97887	0 19823	0 85328	0 89032
preposition |	172	14	1645	9	0 92473	0 95028	0 00844	0 93733	0 97092
pronoun |	34	0	1798	8	1 00000	0 80952	0 00000	0 89474	0 90476
punc |	249	1	1573	17	0 99600	0 93609	0 00064	0 96512	0 96773
adjective |	165	4	1653	18	0 97633	0 90164	0 00241	0 93750	0 94961
conjuction |	68	10	1757	5	0 87179	0 93151	0 00566	0 90066	0 96292
verb |	98	6	1595	141	0.94231	0 41004	0 00375	0 57143	0 70315
interjection |	45	5	1775	15	0 90000	0 75000	0 00281	0 81818	0 87360
adverb |	27	3	1807	3	0 90000	0 90000	0 00166	0 90000	0 94917
num |	20	0	1785	35	1.00000	0 36364	0 00000	0 53333	0 68182
exclamation |	0	0	1839	1	(nan)		0 00000	0 00000	(nan)

Average for the train

F-Score beta=1, microav: 0.846260
F-Score beta=1, macroav: 0.831157
AUC, microav: 0.891110
AUC, macroav: 0.850363
overall accuracy: 0.91060 (1573/1840), of which 1494 exact matches
There were 11 ties of which 4 (36.36%) were correctly resolved

Confusion Matrix:

```
                noun preposition pronoun   punc adjective conjuction   verb interjection adverb    num exclamation
           ---------------------------------------------------------------------------------------------------
   noun |      695       2         0        0       1         7        4         0        1        0       0
preposition | 2        172         0        1       0         1        0         3        2        0       0
pronoun |        7       0        34        0       1         0        0         0        0        0       0
   punc |       16       0         0      249       0         0        0         1        0        0       0
adjective |     17       0         0        0     165         0        1         0        0        0       0
conjuction |     2       2         0        0       1        68        0         0        0        0       0
   verb |      138       1         0        0       0         1       98         1        0        0       0
interjection |6          8         0        0       0         1        0        45        0        0       0
adverb |         2       1         0        0       0         0        0         0       27        0       0
    num |       34       0         0        0       0         0        1         0        0       20       0
exclamation | 0          0         0        0       1         0        0         0        0        0       0
    -*- |        0       0         0        0       0         0        0         0        0        0       0
```

8. *Result of testing with training file train8p and test file tpos8*

Scores per Value Class:

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
noun |	733	204	912	20	0 78228	0 97344	0.18280	0 86746	0 89532
preposition |	228	11	1622	8	0 95397	0 96610	0 00674	0 96000	0 97968
pronoun |	45	0	1816	8	1.00000	0 84906	0 00000	0 91837	0 92453
punc |	242	0	1617	10	1.00000	0 96032	0.00000	0 97976	0 98016
adjective |	137	7	1711	14	0 95139	0 90728	0 00407	0 92881	0 95161
conjuction |	81	11	1773	4	0 88043	0 95294	0 00617	0 91525	0 97339
verb |	66	9	1670	124	0 88000	0 34737	0.00536	0 49811	0 67100
interjection |	35	3	1821	10	0.92105	0.77778	0.00164	0 84337	0 88807
adverb |	29	4	1833	3	0 87879	0 90625	0 00218	0 89231	0 95204
num |	24	0	1797	48	1 00000	0 33333	0 00000	0 50000	0 66667
exclamation |	0	0	1869	0	(nan)		(nan)		0 00000

Average for the train
F-Score beta=1, microav: 0.847108
F-Score beta=1, macroav: 0.830344
AUC, microav: 0.892632
AUC, macroav: 0.888246
overall accuracy: 0.92060 (1620/1869), of which 1543 exact matches
There were 12 ties of which 6 (50.00%) were correctly resolved

Confusion Matrix:
```
              noun preposition pronoun   punc adjective conjuction   verb interjection adverb    num exclamation
   -------------------------------------------------------------------------------------------------------
   noun |        733      0       0       0      5       8       5       0       2       0       0
preposition | 3   228      0       0      2       0       1       2       0       0       0
pronoun |          8      0      45       0      0       0       0       0       0       0       0
   punc |          6      2       0     242      0       1       0       1       0       0       0
adjective |       12      1       0       0    137       0       1       0       0       0       0
conjuction |       0      2       0       0      0      81       0       0       2       0       0
   verb |        123      1       0       0      0       0      66       0       0       0       0
interjection |3            5       0       0      0       2       0      35       0       0       0
adverb |           3      0       0       0      0       0       0       0      29       0       0
    num |         46      0       0       0      0       0       2       0       0      24       0
exclamation | 0           0       0       0      0       0       0       0       0       0       0
    -*- |          0      0       0       0      0       0       0       0       0       0       0
```

9. *Result of testing with training file train9p and test file tpos9*

Scores per Value Class:

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
noun |	739	255	835	19	0.74346	0.97493	0.23394	0.84361	0.87049
preposition |	191	8	1643	6	0.95980	0.96954	0.00485	0.96465	0.98235
pronoun |	32	0	1809	7	1.00000	0.82051	0.00000	0.90141	0.91026
punc |	256	2	1590	0	0.99225	1.00000	0.00126	0.99611	0.99937
adjective |	129	15	1693	11	0.89583	0.92143	0.00878	0.90845	0.95632
conjuction |	37	5	1800	6	0.88095	0.86047	0.00277	0.87059	0.92885
verb |	74	12	1673	89	0.86047	0.45399	0.00712	0.59438	0.72343
interjection |	35	2	1802	9	0.94595	0.79545	0.00111	0.86420	0.89717
adverb |	29	0	1815	4	1.00000	0.87879	0.00000	0.93548	0.93939
num |	27	0	1674	147	1.00000	0.15517	0.00000	0.26866	0.57759
exclamation |	0	0	1847	1	(nan)		0.00000	0.00000	(nan)

Average for the train
F-Score beta=1, microav: 0.842812
F-Score beta=1, macroav: 0.814753
AUC, microav: 0.888554
AUC, macroav: 0.844112
overall accuracy: 0.92690 (1549/1848), of which 1420 exact matches
There were 11 ties of which 2 (18.18%) were correctly resolved

Confusion Matrix:
```
         noun preposition pronoun   punc adjective conjuction   verb interjection adverb    num exclamation
   -------------------------------------------------------------------------------------------------------
   noun |   739      0       0       0      7       4       8       0       0       0       0
preposition | 3   191      0       0      0       0       1       2       0       0       0
pronoun |     6      1      32       0      0       0       0       0       0       0       0
```

```
     punc |        0       0       0    256      0       0       0       0       0       0       0
adjective |       11       0       0      0    129       0       0       0       0       0       0
conjuction |       0       2       0      0      4      37       0       0       0       0       0
     verb |       87       1       0      0      1       0      74       0       0       0       0
interjection |2            3       0      1      2       1       0      35       0       0       0
   adverb |        0       1       0      0      1       0       2       0      29       0       0
      num |      146       0       0      0      0       0       1       0       0      27       0
exclamation | 0            0       0      1      0       0       0       0       0       0       0
      -*- |        0       0       0      0      0       0       0       0       0       0       0
```

10. Result of testing with training file train10p and test file tpos10

Scores per Value Class

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
noun |	667	231	925	25	0.74276	0.96387	0.19983	0.83899	0.88202
preposition |	240	8	1588	12	0.96774	0.95238	0.00501	0.96000	0.97368
pronoun |	36	2	1806	4	0.94737	0.90000	0.00111	0.92308	0.94945
punc |	250	1	1594	3	0.99602	0.98814	0.00063	0.99206	0.99376
adjective |	163	5	1668	12	0.97024	0.93143	0.00299	0.95044	0.96422
conjuction |	50	8	1785	5	0.86207	0.90909	0.00446	0.88496	0.95231
verb |	82	13	1584	169	0.86316	0.32669	0.00814	0.47399	0.65928
interjection |	49	7	1786	6	0.87500	0.89091	0.00390	0.88288	0.94350
adverb |	27	3	1815	3	0.90000	0.90000	0.00165	0.90000	0.94917
num |	6	0	1803	39	1.00000	0.13333	0.00000	0.23529	0.56667
exclamation |	0	0	1848	0	(nan)		(nan)		0.00000

Average for the train

F-Score beta=1, microav:	0.826820
F-Score beta=1, macroav:	0.804169
AUC, microav:	0.885841
AUC, macroav:	0.883407
overall accuracy:	0.887400 (1570/1848), of which 1501 exact matches

There were 13 ties of which 6 (46.15%) were correctly resolved

Confusion Matrix:

```
             noun preposition pronoun  punc adjective conjuction  verb interjection adverb   num
exclamation
------------------------------------------------------------------------------------------------
    noun |     667      2       0      0      4       8      10       1       0       0       0
preposition | 7       240       0      0      0       0       0       5       0       0       0
 pronoun |       4      0      36      0      0       0       0       0       0       0       0
    punc |       2      0       0    250      0       0       0       1       0       0       0
adjective |     10      0       2      0    163       0       0       0       0       0       0
conjuction |     2      2       0      0      0      50       0       0       1       0       0
    verb |     164      1       0      1      1       0      82       0       2       0       0
interjection |3         3       0      0      0       0       0      49       0       0       0
  adverb |       0      0       0      0      0       0       3       0      27       0       0
     num |      39      0       0      0      0       0       0       0       0       6       0
exclamation | 0         0       0      0      0       0       0       0       0       0       0
     -*- |       0      0       0      0      0       0       0       0       0       0       0
```

PART TWO: NAME ENTITY RECOGNIZER EVALUATION DATA.

1. ***Result of testing with training file train1n and test file tner1***

Scores per Value Class

class	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
O |	1585	8	250	4	0.99498	0.99748	0.03101	0.99623	0.98324
I-PER |	146	6	1690	5	0.96053	0.96689	0.00354	0.96370	0.98167
I-LOC |	68	5	1767	7	0.93151	0.90667	0.00282	0.91892	0.95192
B-LOC |	0	0	1845	2	(nan)		0.00000	0.00000	(nan)
I-ORG |	16	3	1828	0	0.84211	1.00000	0.00164	0.91429	0.99918
B-ORG |	7	3	1834	3	0.70000	0.70000	0.00163	0.70000	0.84918
B-PER |	0	0	1845	2	(nan)		0.00000	0.00000	(nan)
num |	0	0	1847	0	(nan)		(nan)		0.00000

Average for the train
F-Score beta=1, microav: 0.989586
F-Score beta=1, macroav: 0.898626
AUC, microav: 0.980516
AUC, macroav: 0.783150
overall accuracy: 98.07% (1822/1847), of which 1590 exact matches
There were 2 ties of which 1 (50.00%) were correctly resolved

Confusion Matrix:

```
             O  I-PER  I-LOC  B-LOC  I-ORG  B-ORG  B-PER    num
      -----------------------------------------------------------------
    O |   1585      1      2      0      0      1      0      0
I-PER |      5    146      0      0      0      0      0      0
I-LOC |      0      4     68      0      3      0      0      0
B-LOC |      0      0      0      0      0      2      0      0
I-ORG |      0      0      0      0     16      0      0      0
B-ORG |      0      0      3      0      0      7      0      0
B-PER |      1      1      0      0      0      0      0      0
  num |      0      0      0      0      0      0      0      0
  -*- |      1      0      0      0      0      0      0      0
```

2. ***Result of testing with training file train2n and test file tner2***

Scores per Value Class

class	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
O |	1651	48	148	2	0.97175	0.99879	0.24490	0.98508	0.87695
I-PER |	61	1	1767	20	0.98387	0.75309	0.00057	0.85315	0.87626
I-LOC |	52	6	1768	23	0.89655	0.69333	0.00338	0.78195	0.84498
I-ORG |	16	4	1825	4	0.80000	0.80000	0.00219	0.80000	0.89891
B-ORG |	5	1	1841	2	0.83333	0.71429	0.00054	0.76923	0.85687
B-PER |	0	0	1843	6	(nan)		0.00000	0.00000	(nan)
B-LOC |	3	0	1842	4	1.00000	0.42857	0.00000	0.60000	0.71429
num |	0	1	1848	0	0.00000	(nan)		0.00054	(nan)

Average for the train
F-Score beta=1, microav: 0.965244
F-Score beta=1, macroav: 0.798236
AUC, microav: 0.875335

AUC, macroav: 0.795464
overall accuracy: 96.21% (1788/1849), of which 1497 exact matches
There was 1 tie of which 1 (100.00%) was correctly resolved

Confusion Matrix:

```
           O  I-PER  I-LOC  I-ORG  B-ORG   B-PER   B-LOC    num

     -------------------------------------------------------------------------
    O |  1651     0     1      0     0      0      0      1
I-PER |    20    61     0      0     0      0      0      0
I-LOC |    18     1    52      4     0      0      0      0
I-ORG |     4     0     0     16     0      0      0      0
B-ORG |     0     0     2      0     5      0      0      0
B-PER |     6     0     0      0     0      0      0      0
B-LOC |     0     0     3      0     1      0      3      0
  num |     0     0     0      0     0      0      0      0
  -*- |     0     0     0      0     0      0      0      0
```

3. ***Result of testing with training file train3n and test file tner3***

Scores per Value Class:

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
O |	1642	39	156	5	0.97680	0.99696	0.20000	0.98678	0.89848
I-PER |	78	3	1745	16	0.96296	0.82979	0.00172	0.89143	0.91404
I-LOC |	55	1	1772	14	0.98214	0.79710	0.00056	0.88000	0.89827
I-ORG |	11	1	1827	3	0.91667	0.78571	0.00055	0.84615	0.89258
B-ORG |	4	1	1837	0	0.80000	1.00000	0.00054	0.88889	0.99973
B-PER |	0	1	1841	0	0.00000	(nan)		0.00054	(nan)
B-LOC |	6	0	1834	2	1.00000	0.75000	0.00000	0.85714	0.87500
num |	0	0	1836	6	(nan)		0.00000	0.00000	(nan)

Average for the train

F-Score beta=1, microav: 0.974000
F-Score beta=1, macroav: 0.891732
AUC, microav: 0.899876
AUC, macroav: 0.854014
overall accuracy: 97.56% (1796/1842), of which 1499 exact matches
There were 4 ties of which 3 (75.00%) were correctly resolved

Confusion Matrix:

```
           O    I-PER   I-LOC  I-ORG  B-ORG  B-PER   B-LOC    num

     -------------------------------------------------------------------------
    O |  1642     2      1      0      1      1      0      0
I-PER |    16    78      0      0      0      0      0      0
I-LOC |    12     1     55      1      0      0      0      0
I-ORG |     3     0      0     11      0      0      0      0
B-ORG |     0     0      0      0      4      0      0      0
B-PER |     0     0      0      0      0      0      0      0
B-LOC |     2     0      0      0      0      0      6      0
  num |     6     0      0      0      0      0      0      0
  -*- |     0     0      0      0      0      0      0      0
```

4. ***Result of testing with training file train4n and test file tner4***

Scores per Value Class

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
O |	1631	25	200	6	0.98490	0.99633	0.11111	0.99059	0.94261
I-PER |	117	3	1728	14	0.97500	0.89313	0.00173	0.93227	0.94570
I-LOC |	57	1	1798	6	0.98276	0.90476	0.00056	0.94215	0.95210
I-ORG |	14	3	1844	1	0.82353	0.93333	0.00162	0.87500	0.96585
B-ORG |	8	1	1852	1	0.88889	0.88889	0.00054	0.88889	0.94417
B-PER |	0	0	1858	4	(nan)		0.00000	0.00000	(nan)
B-LOC |	1	0	1860	1	1.00000	0.50000	0.00000	0.66667	0.75000
num |	0	1	1860	1	0.00000	0.00000	0.00054	(nan)	

Average for the train
F-Score beta=1, microav: 0.982777
F-Score beta=1, macroav: 0.882594
AUC, microav: 0.942285
AUC, macroav: 0.812522
overall accuracy: 98.07% (1828/1862), of which 1580 exact matches
There was 1 tie of which 1 (100.00%) was correctly resolved

Confusion Matrix:

```
          O   I-PER   I-LOC   I-ORG  B-ORG   B-PER  B-LOC     num
-----------------------------------------------------------------------
    O |  1631     3       0       2      0       0      0       1
I-PER |    14   117       0       0      0       0      0       0
I-LOC |     5     0      57       1      0       0      0       0
I-ORG |     1     0       0      14      0       0      0       0
B-ORG |     0     0       1       0      8       0      0       0
B-PER |     4     0       0       0      0       0      0       0
B-LOC |     0     0       0       0      1       0      1       0
  num |     1     0       0       0      0       0      0       0
  -*- |     0     0       0       0      0       0      0       0
```

5. ***Result of testing with training file train5n and test file tner5***

Scores per Value Class

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
O |	1643	40	181	4	0.97623	0.99757	0.18100	0.98679	0.90829
I-PER |	96	1	1750	21	0.98969	0.82051	0.00057	0.89720	0.90997
I-LOC |	53	3	1797	15	0.94643	0.77941	0.00167	0.85484	0.88887
I-ORG |	22	0	1844	2	1.00000	0.91667	0.00000	0.95652	0.95833
B-ORG |	6	0	1862	0	1.00000	1.00000	0.00000	1.00000	1.00000
B-PER |	0	2	1862	4	0.00000	0.00000	0.00107	(nan)	
B-LOC |	2	0	1866	0	1.00000	1.00000	0.00000	1.00000	1.00000
num |	0	0	1868	0	(nan)		(nan)		0.00000

Average for the train
F0 -Score beta=1, microav: 0.975164
F-Score beta=1, macroav: 0.949224
AUC, microav: 0.908000
AUC, macroav: 0.880704
overall accuracy: 97.48% (1822/1868), of which 1547 exact matches
There were 3 ties of which 1 (33.33%) were correctly resolved

Confusion Matrix:

```
         O  I-PER    I-LOC  I-ORG  B-ORG   B-PER B-LOC     num
-----------------------------------------------------------------------
```

```
      O |   1643      1      1      0      0      2      0      0
  I-PER |     21     96      0      0      0      0      0      0
  I-LOC |     15      0     53      0      0      0      0      0
  I-ORG |      0      0      2     22      0      0      0      0
  B-ORG |      0      0      0      0      6      0      0      0
  B-PER |      4      0      0      0      0      0      0      0
  B-LOC |      0      0      0      0      0      0      2      0
    num |      0      0      0      0      0      0      0      0
    -*- |      0      0      0      0      0      0      0      0
```

6. ***Result of testing with training file train6n and test file tner6***

Scores per Value Class:

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
O |	1692	20	188	0	0 98832	1.00000	0.09615	0 99412	0 95192
I-PER |	90	0	1798	12	1.00000	0 88235	0 00000	0.93750	0 94118
I-LOC |	71	2	1821	6	0 97260	0 92208	0.00110	0.94667	0.96049
I-ORG |	17	0	1883	0	1.00000	1.00000	0.00000	1 00000	1.00000
B-ORG |	6	0	1892	2	1.00000	0.75000	0 00000	0 85714	0.87500
B-PER |	0	0	1898	2	(nan)		0 00000	0 00000	(nan)
B-LOC |	2	0	1898	0	1.00000	1.00000	0 00000	1.00000	1.00000
num |	0	0	1900	0	(nan)		(nan)		0 00000

Average for the train
F-Score beta=1, microav: 0.987677
F-Score beta=1, macroav: 0.955906
AUC, microav: 0.950877
AUC, macroav: 0.889799
overall accuracy: 98.58% (1878/1900), of which 1613 exact matches
There were 2 ties of which 2 (100.00%) were correctly resolved

Confusion Matrix:

```
             O  I-PER  I-LOC  I-ORG  B-ORG  B-PER  B-LOC    num
-------------------------------------------------------------------------
      O |   1692      0      0      0      0      0      0      0
  I-PER |     12     90      0      0      0      0      0      0
  I-LOC |      6      0     71      0      0      0      0      0
  I-ORG |      0      0      0     17      0      0      0      0
  B-ORG |      0      0      2      0      6      0      0      0
  B-PER |      2      0      0      0      0      0      0      0
  B-LOC |      0      0      0      0      0      0      2      0
    num |      0      0      0      0      0      0      0      0
    -*- |      0      0      0      0      0      0      0      0
```

7. ***Result of testing with training file train7n and test file tner7***

Scores per Value Class:

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
O |	1685	22	135	0	0 98711	1.00000	0.14013	0.99351	0 92994

I-PER |	79	0	1748	15	1.00000	0.84043	0.00000	0.91329	0.92021
I-LOC |	36	0	1804	2	1.00000	0.94737	0.00000	0.97297	0.97368
I-ORG |	17	0	1822	3	1.00000	0.85000	0.00000	0.91892	0.92500
B-ORG |	3	0	1839	0	1.00000	1.00000	0.00000	1.00000	1.00000
B-PER |	0	0	1841	1	(nan)		0.00000	0.00000	(nan)
B-LOC |	0	0	1842	0	(nan)		(nan)		0.00000
num |	0	0	1841	1	(nan)		0.00000	0.00000	(nan)

Average for the train

F-Score beta=1, microav: 0.985813
F-Score beta=1, macroav: 0.959740
AUC, microav: 0.929982
AUC, macroav: 0.821262
overall accuracy: 98.64% (1820/1842), of which 1490 exact matches
There was 1 tie of which 0 (0.00%) was correctly resolved

Confusion Matrix:

```
          O     I-PER  I-LOC  I-ORG  B-ORG   B-PER B-LOC    num
      -------------------------------------------------------------------------------
    O |  1685     0      0      0      0      0      0      0
I-PER |    15    79      0      0      0      0      0      0
I-LOC |     2     0     36      0      0      0      0      0
I-ORG |     3     0      0     17      0      0      0      0
B-ORG |     0     0      0      0      3      0      0      0
B-PER |     1     0      0      0      0      0      0      0
B-LOC |     0     0      0      0      0      0      0      0
  num |     1     0      0      0      0      0      0      0
  -*- |     0     0      0      0      0      0      0      0
```

8. *Result of testing with training file train8n and test file tner8*

Scores per Value Class:

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
O |	1619	24	188	2	0.98539	0.99877	0.11321	0.99203	0.94278
I-PER |	118	3	1697	15	0.97521	0.88722	0.00176	0.92913	0.94273
I-LOC |	52	1	1774	6	0.98113	0.89655	0.00056	0.93694	0.94799
I-ORG |	12	0	1821	0	1.00000	1.00000	0.00000	1.00000	1.00000
B-ORG |	4	0	1828	1	1.00000	0.80000	0.00000	0.88889	0.90000
B-PER |	0	0	1830	3	(nan)		0.00000	0.00000	(nan)
B-LOC |	0	0	1833	0	(nan)		(nan)		0.00000
num |	0	0	1832	1	(nan)		0.00000	0.00000	(nan)

Average for the train

F-Score beta=1, microav: 0.985012
F-Score beta=1, macroav: 0.949399
AUC, microav: 0.942401
AUC, macroav: 0.819071

overall accuracy: 98.31% (1805/1833), of which 1511 exact matches

Confusion Matrix:

```
          O  I-PER    I-LOC  I-ORG  B-ORG     B-PER B-LOC    num
     ---------------------------------------------------------------------------------------
    O |  1619     2     0     0     0     0     0     0
I-PER |    15   118     0     0     0     0     0     0
I-LOC |     6     0    52     0     0     0     0     0
I-ORG |     0     0     0    12     0     0     0     0
B-ORG |     0     0     1     0     4     0     0     0
B-PER |     2     1     0     0     0     0     0     0
B-LOC |     0     0     0     0     0     0     0     0
  num |     1     0     0     0     0     0     0     0
  -*- |     0     0     0     0     0     0     0     0
```

9. *Result of testing with training file train9n and test file tner9*

Scores per Value Class:

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
O |	1612	30	202	1	0.98173	0.99938	0.12931	0.99048	0.93503
I-PER |	121	5	1713	6	0.96032	0.95276	0.00291	0.95652	0.97492
I-LOC |	61	1	1760	23	0.98387	0.72619	0.00057	0.83562	0.86281
I-ORG |	10	0	1835	0	1.00000	1.00000	0.00000	1.00000	1.00000
B-ORG |	5	0	1839	1	1.00000	0.83333	0.00000	0.90909	0.91667
B-PER |	0	0	1841	4	(nan)		0.00000	0.00000	(nan)
B-LOC |	0	0	1845	0	(nan)		(nan)		0.00000
num |	0	0	1844	1	(nan)		0.00000	0.00000	(nan)

Average for the train

F-Score beta=1, microav:	0.982605
F-Score beta=1, macroav:	0.938341
AUC, microav:	0.935429
AUC, macroav:	0.812777
overall accuracy:	96.75% (1809/1845), of which 1418 exact matches

There was 1 tie of which 1 (100.00%) was correctly resolved

Confusion Matrix:

```
          O   I-PER  I-LOC  I-ORG  B-ORG B-PER  B-LOC    num
     ---------------------------------------------------------------------------------------
    O |  1612     1     0     0     0     0     0     0
I-PER |     6   121     0     0     0     0     0     0
I-LOC |    23     0    61     0     0     0     0     0
I-ORG |     0     0     0    10     0     0     0     0
B-ORG |     0     0     1     0     5     0     0     0
B-PER |     0     4     0     0     0     0     0     0
B-LOC |     0     0     0     0     0     0     0     0
  num |     1     0     0     0     0     0     0     0
  -*- |     0     0     0     0     0     0     0     0
```

10. *Result of testing with training file train10n and test file tner10*

Scores per Value Class

class |	TP	FP	TN	FN	precision	recall(TPR)	FPR	F-score	AUC
O |	1655	25	160	0	0.98512	1.00000	0.13514	0.99250	0.93243

I-PER |	107	1	1712	20	0.99074	0.84252	0.00058	0.91064	0.92097
I-LOC |	35	1	1801	3	0.97222	0.92105	0.00055	0.94595	0.96025
I-ORG |	12	0	1828	0	1.00000	1.00000	0.00000	1.00000	1.00000
B-ORG |	4	0	1835	1	1.00000	0.80000	0.00000	0.88889	0.90000
B-PER |	0	0	1838	2	(nan)		0.00000	0.00000	(nan)
B-LOC |	0	0	1840	0	(nan)		(nan)		0.00000
num |	0	0	1839	1	(nan)		0.00000	0.00000	(nan)

Average for the train

F-Score beta=1, microav:	0.984212
F-Score beta=1, macroav:	0.947595
AUC, microav:	0.932089
AUC, macroav:	0.816236
overall accuracy:	98.42% (1813/1840), of which 1493 exact matches

There were 2 ties of which 2 (100.00%) were correctly resolved

Confusion Matrix:

```
           O      I-PER  I-LOC  I-ORG  B-ORG  B-PER  B-LOC   num
      ------------------------------------------------------------
     O |  1655      0      0      0      0      0      0      0
 I-PER |    20    107      0      0      0      0      0      0
 I-LOC |     3      0     35      0      0      0      0      0
 I-ORG |     0      0      0     12      0      0      0      0
 B-ORG |     0      0      1      0      4      0      0      0
 B-PER |     1      1      0      0      0      0      0      0
 B-LOC |     0      0      0      0      0      0      0      0
   num |     1      0      0      0      0      0      0      0
   -*- |     0      0      0      0      0      0      0      0
```

Appendix B Resources and Time schedule

Resources Required:

TYPE OF ITEM	COST
Laptop	Ksh 70 000
Kĩkamba dictionary	Ksh 1500
Timbl software	Free
Sun xv virtual machine	Free
Traveling(collecting corpus)	Ksh 30 000
Printing and typing	Ksh 5 000
Information(corpus)	Ksh 10 000
TOTAL	KSH 116,500

Schedule

Activity	Duration (weeks)	Start week	End week
Literature review/problem formulation	3	1	3
Proposal writing	1	4	4
Proposal presentation	2	5	6
Corpus collection	4	7	10
System analysis and design/annotation	4	11	14
Application development	4	15	18
Application demonstration	2	19	20
Test & discuss result	3	21	23
Report writing	1	24	24
Project Presentation	2	25	26

Appendix C Tagger code

```
#advanced Tkinter user interface
#28/nov/2010

import os
from Tkinter import *


class Application:
    def __init__(self, master):
        frame = Frame(master, width=500, height=400, bd=1)
        frame.pack()

        self.frame1 = Frame(frame, relief = 'flat', bd=2)
        self.frame1.pack(fill = X)

        #create frame 1 text
        self.heading_lbl = Label(self.frame1, text = "Input Kamba text separating word and  punctuation mark
by space")
        self.heading_lbl.grid(row=1, column=0, sticky=W)

        #create frame 2
        self.frame2 = Frame(frame, relief = 'flat', bd=2)
        self.frame2.pack(fill = X)

        self.userquestion_txt = Text(self.frame2, width = 69, height = 5, wrap = WORD, background =
'#FFFFCC')
        self.userquestion_txt.grid(row = 3, column = 1, columnspan = 2, sticky = W)

        #create frame 3
        self.frame3 = Frame(frame, relief = 'flat', bd=2)
        self.frame3.pack(fill = X)

        Button(self.frame3, text='Submit', command = self.reveal).pack(side=LEFT, padx=5)
        Button(self.frame3, text='Clear', command = self.clear_all_text).pack(side=RIGHT, padx=5)

        #create frame 4
        self.frame4 = Frame(frame, relief = 'flat', bd=2)
        self.frame4.pack(fill = X)

        self.frame4_lbl1 = Label(self.frame4, text = "part of speech")
        self.frame4_lbl1.grid(row=0, column=0, sticky=W)

        self.nameentityrecorgnizer_txt = Text(self.frame4, width = 34, height = 18, wrap = WORD,
background = '#FFFFCC')
        self.nameentityrecorgnizer_txt.grid(row = 1, column = 0, columnspan = 2, sticky = W)

        self.frame4_lbl2 = Label(self.frame4, text = "Name entity recognizer")
        self.frame4_lbl2.grid(row=0, column=2, sticky=W)

        self.partsofspeech_txt = Text(self.frame4, width = 34, height = 18, wrap = WORD, background =
'#FFFFCC')
        self.partsofspeech_txt.grid(row = 1, column = 2, columnspan = 2, sticky = W)
```

```
 def clear_all_text(self):
    """Clear All text boxes"""
    self.userquestion_txt.delete(0.0, END)
    self.nameentityrecorgnizer_txt.delete(0.0, END)
    self.partsofspeech_txt.delete(0.0, END)

 def reveal(self):
    """Display message based on input text """
    fileobj = open('syst001.txt','w')
    try:
       texttoparse = self.userquestion_txt.get(0.0, END)
       fileobj.write(texttoparse)
    except:
       self.nameentityrecorgnizer_txt.insert(END, 'failed to save file !')

    fileobj.close()

    contents = "Mbt -s testgen.settings  -t "+'syst001.txt'
    contents2 = "Mbt -s nergen.settings  -t "+'syst001.txt'

    if contents != "":
       #message= "There is text to parse "+contents
       try:
          message = os.popen(contents).read()
       except:
          message = "Failed to execute testgen.settings"

       #message2
       try:
          message2 = os.popen(contents2).read()
       except:
          message2 = "Failed to execute nergen.settings"

    else:
       message="There is no text to parse"

    self.nameentityrecorgnizer_txt.delete(0.0, END)
    self.nameentityrecorgnizer_txt.insert(0.0, message)

    self.partsofspeech_txt.delete(0.0, END)
    self.partsofspeech_txt.insert(0.0, message2)



root = Tk()
root.geometry("500x400")
root.title("KAMBA NAME ENTITY RECOGNIZER & PART OF SPEECH")
root.option_add('*font', ('verdana', 10, 'bold'))

app = Application(root)
root.mainloop()
```