A BACHELOR'S MINI PROJECT ON

Unsupervised Word Sense Disambiguation

Submitted in partial fulfillment

Of the requirements for the completion of the 6TH semester

Of the

UNDER GRADUATE PROGRAM

In

INFORMATION TECHNOLOGY
(B.Tech in IT)

Submitted by :

1. ***Vikhyat Tandon (IIT2011203)***
2. ***Vikash Kumar (IIT2011209)***
3. ***Shubham Mishra (IIT2011211)***

Under the Guidance of:

Prof. U.S. Tiwary
IIIT-Allahabad

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY
ALLAHABAD – 211012 (INDIA)
October, 2013 - 2014

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this project entitled "**Unsupervised Word Sense Disambiguation**", submitted in the partial fulfillment of the completion of the 6th semester of Bachelor of Technology (B.Tech) program, in Information Technology at Indian Institute of Information Technology, Allahabad, is an authentic record of my original work carried out under the guidance of **Prof. U. S. Tiwary** due acknowledgements have been made in the text of the project to all other material used. This semester work was done in full compliance with the requirements and constraints of the prescribed curriculum.

Place: Allahabad
Date: 12th May, 2014

1. Vikhyat Tandon (IIT2011203)
2. Vikash Kumar (IIT2011209)
3. Shubham Mishra (IIT2011211)
(B.Tech IIIrd year)

CERTIFICATE FROM SUPERVISOR

I do hereby certify that the mini project report prepared under my supervision by ***Vikhyat Tandon(IIT2011203)*** , ***Vikash Kumar(IIT2011209)*** and ***Shubham Mishra(IIT2011211)*** titled "Unsupervised Word Sense Disambiguation" be **accepted** in the partial fulfillment of the requirements of the completion of 6th semester of Bachelor of Technology in Information Technology **for Examination.**

Date: 12th May, 2014 **Prof. U. S. Tiwary**

Place: Allahabad IIIT- Allahabad

ACKNOWLEDGEMENTS

The author would like to express his sincere gratitude to our project guide **Prof. U. S. Tiwary** for guiding us throughout the course of the project. We are highly indebted to him for his invaluable guidance and ever-ready support. His deep knowledge of computer engineering field made us realize that theoretical knowledge always helps develop efficient operational industrial software, which are a blend of all core subjects of the field. Working under his guidance has been a fruitful experience, which will be very valuable for us, when we enter the corporate world.

We would like to give a warm expression of thanks to honorable **Director Sir** for providing the facilities and academic environment for our project work.

Place: Allahabad
Date : 12th May, 2014

1. **Vikhyat Tandon (IIT2011203)**
2. **Vikash Kumar (IIT2011209)**
3. **Shubham Mishra (IIT2011211)**

(**B.Tech IIIrd Year)**

Abstract

Word sense disambiguation (WSD) is the task of selecting the appropriate senses of a word in a given context. It is essence of communication in a natural language. It is motivated by its use in many crucial applications such as Information retrieval, Information extraction, Machine translation, Part of-Speech tagging, etc. Various issues like scalability, ambiguity, diversity (of languages) and evaluation pose challenges to WSD solutions. The aim of this project is to develop a unsupervised WSD technique which can handle all these issues with better accuracy and performance. This report presents our preliminary work towards solving the problem.

Table of Contents

Introduction:

"When I use a word," Humpty Dumpty said,...
"it means just what I choose it to mean – neither more nor less."
-Lewis Carroll (1875)

Words can have different senses. Some words have multiple meanings. This is called Polysemy. For example: bank can be a financial institute or a river shore. Sometimes two completely different word are spelled the same. For example: Can, can be used as model verb: You can do it, or as container: She brought a can of soda. This is called Homonymy. Distinction between polysemy and homonymy is not always clear. Word sense disambiguation (WSD) is the problem of determining in which sense a word having a number of distinct senses is used in a given sentence. Take another example, consider the word "bass", with two distinct senses:

1. a type of fish
2. tones of low frequency

and the sentences "The bass part of the song is very moving" and "I went fishing for some sea bass". To a human it is obvious the first sentence is using the word "bass" in sense 2 above, and in the second sentence it is being used in sense 1.But although this seems obvious to a human, developing algorithms to replicate this human ability is a difficult task. One problem with word sense disambiguation is deciding what are the senses. In cases like the word "bass" above, at least some senses are obviously different. In other cases, however, the different senses can be closely related (one meaning being a metaphorical extension of another), and in such cases division of words into senses becomes much more difficult. Consulting different dictionaries will find many different divisions of words into senses. One solution some researchers have used is to choose a particular dictionary, and just use its set of senses. There are two main approaches to WSD – deep approaches and shallow approaches. Deep approaches presume access to a comprehensive body of world knowledge. Knowledge such as "you can go fishing for a type of fish, but not for low frequency sounds" and "songs have low frequency sounds as parts, but not types of fish" is then used to determine in which sense the word is used. These approaches are not very successful in practice, mainly because we don't have access to such a body of knowledge, except in very limited domains. But if such knowledge did exist, they would be much better than the shallow approaches. Shallow approaches don't try to understand the text. They just consider the surrounding words, using information like "if 'bass' has words 'sea' or 'fishing' nearby, it probably is in the fish sense; if 'bass' has the words 'music' or 'song' nearby, it is probably in the music sense." These rules can be automatically derived by the computer, using a training corpus of words tagged with their word senses. This approach, while theoretically not as powerful as deep approaches, gives superior results in practice, due to our limited world knowledge. It can, though, be confused by sentences like "The dog barked at the tree."

This report highlights main issues in word sense disambiguation, various approaches and solutions proposed till date.

Motivation:

Word sense disambiguation a task of removing the ambiguity of word in context, is important for many NLP applications such as:

- Information Retrieval: As proposed by WSD helps in improving term indexing in information retrieval has proved that word senses improve retrieval performance if the senses are included as index terms. Thus, documents should not be ranked based on words alone, the documents should be ranked based on word senses, or based on a combination of word senses and words. For example: Using different indexes for keyword “Java” as “programming language”, as “type of coffee”, and as “location” will improve accuracy of an IR system. Apart from indexing, WSD also helps in query expansion. Short queries are expanded using words that belong to same sync-sets. Retrieval using expanded queries gives better results than original queries. Thus, WSD is crucial for improving accuracy of IR as it eliminates irrelevant hits.
- Machine Translation: WSD is important for Machine translations. It helps in better understanding of source language and generation of sentences in target language. It also affects lexical choice depending upon the usage context.
- Speech Processing and Part of Speech tagging: Speech recognition i.e, when processing homophones words which are spelled differently but pronounced the same way. For example: “base” and “bass” or “sealing” and “ceiling”.
- Text Processing: Text to Speech translation i.e, when words are pronounced in more than one way depending on their meaning. For example: “lead” can be “in front of” or “type of metal”.

Problem Definition:

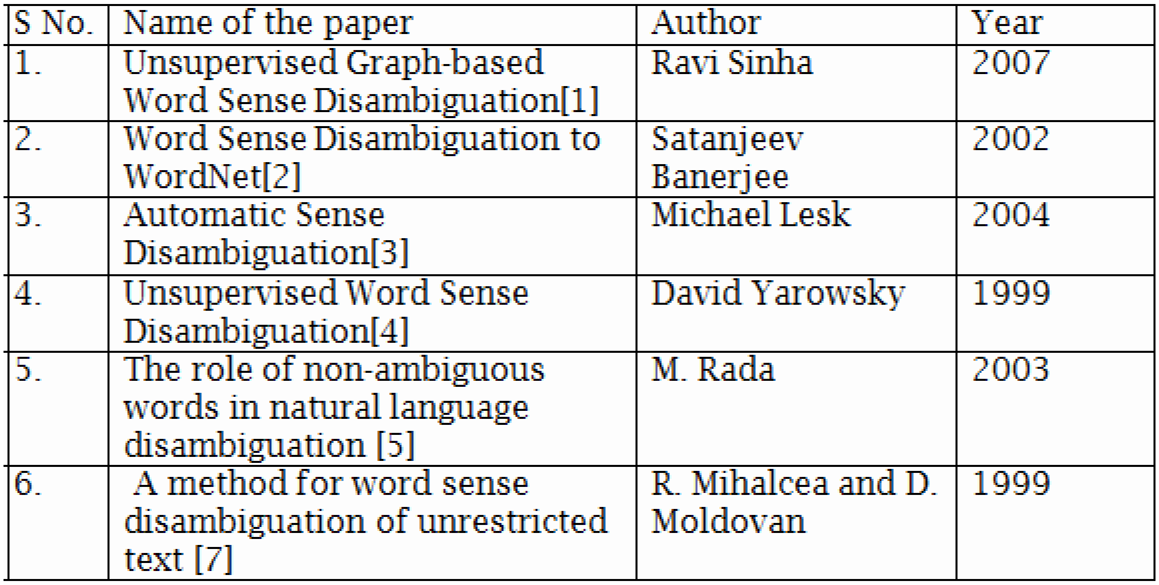
Word sense disambiguation (WSD) involves the association of a given word in a text or discourse with a definition or meaning which is distinguishable from other meanings potentially attributable to that word. The task therefore necessarily involves two steps according to Ide and Veronis (1998). The first step is to deter-mine all the different senses for every word relevant to the text or discourse under consideration, i.e., to choose a sense inventory, e.g., from the lists of senses in everyday dictionaries, from the synonyms in a thesaurus, or from the translations in a translation dictionary.

The second step involves a means to assign the appropriate sense to each occurrence of a word in context. All disambiguation work involves matching the context of an instance of the word to be disambiguated either with information from external knowledge sources or with contexts of previously disambiguated in-stances of the word. For both of these sources we need preprocessing or knowledge-extraction procedures representing the information as context features. For some disambiguation tasks, there are already well-known procedures such as morpho-syntactic disambiguation and therefore WSD has largely focused on distinguishing senses among homographs belonging to the same syntactic category.

Finally a third step is also involved: the computer needs to learn how to associate a word sense with a word in context using either machine learning or manual creation of rules or metrics. Main focus of this report is the use of machine learning approaches for WSD. In these approaches, systems are trained to perform the task of word sense disambiguation. In these approaches first a classifier is learned from the training examples, which is later used to assign senses to unseen examples.

Literature Survey:

Books and research papers studied and referred are as follows:



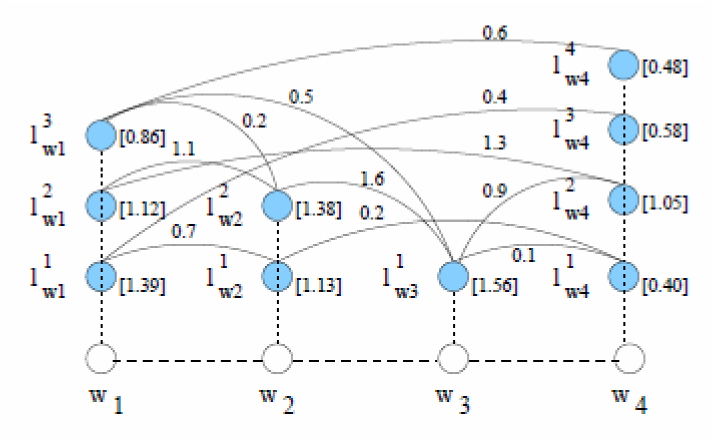
S No.	Name of the paper	Author	Year
1.	Unsupervised Graph-based Word Sense Disambiguation[1]	Ravi Sinha	2007
2.	Word Sense Disambiguation to WordNet[2]	Satanjeev Banerjee	2002
3.	Automatic Sense Disambiguation[3]	Michael Lesk	2004
4.	Unsupervised Word Sense Disambiguation[4]	David Yarowsky	1999
5.	The role of non-ambiguous words in natural language disambiguation [5]	M. Rada	2003
6.	A method for word sense disambiguation of unrestricted text [7]	R. Mihalcea and D. Moldovan	1999

Proposed Approach:

We are using **Unsupervised approach** for Word sense disambiguation. Unsupervised approaches to sense disambiguation make the use of sense tagged data of any kind during the training. In this technique, feature vector representations of unlabeled instances are taken as input and are then grouped into clusters according to a similarity metric. These clusters are then labeled by hand with known word senses. Main disadvantage is that senses are not well defined. We are using the **Graph-based** method for word sense disambiguation

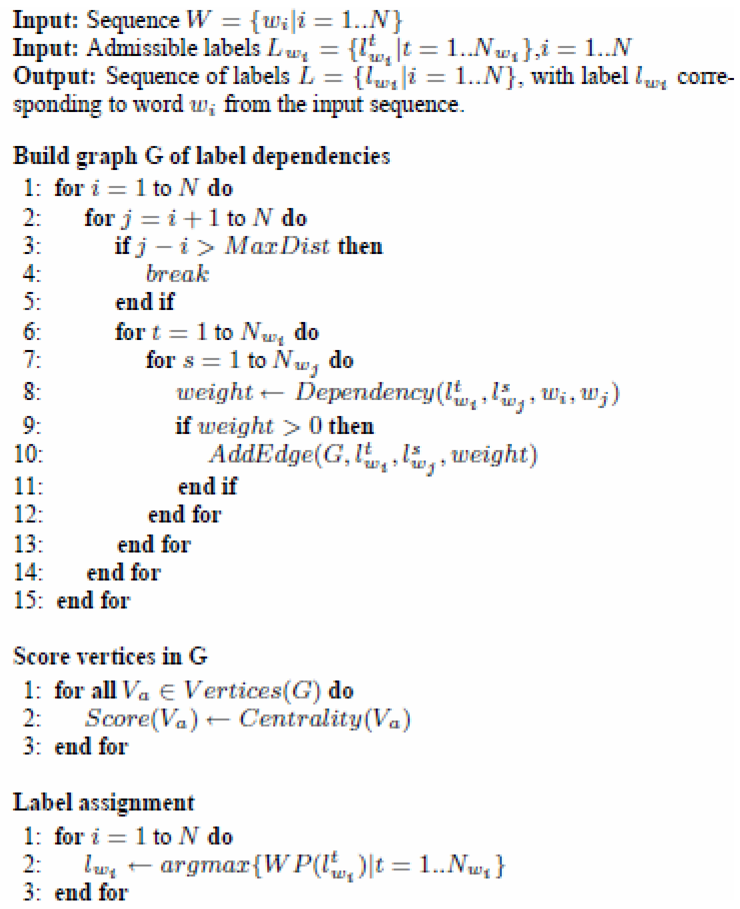
Graph-based Centrality for Word Sense Disambiguation:

Given a sequence of words W = {w1,w2, ...,wn},each word with corresponding admissible labels, we define a label graph G = (V,E) such that there is a vertex v belonging to V for every possible label . Dependencies between pairs of labels are represented as directed or in directed edges e belonging to E, defined over the set of vertex pairs V ×V . Such label dependencies can be learned from annotated data, or derived by other means.



Sample graph built on the set of possible labels (shaded nodes) for a sequence of four words (unshaded nodes).Label dependencies are indicated as edge weights. Scores computed by the graph based algorithm are shown in brackets, next to each label.

Graph Centrality for Word Sense Disambiguation :



```
**Input:** Sequence $W = \{w_i | i = 1..N\}$
**Input:** Admissible labels $L_{w_i} = \{l^t_{w_i} | t = 1..N_{w_i}\}, i = 1..N$
**Output:** Sequence of labels $L = \{l_{w_i} | i = 1..N\}$, with label $l_{w_i}$ corresponding to word $w_i$ from the input sequence.

**Build graph G of label dependencies**
for $i = 1$ to $N$ do
    for $j = i + 1$ to $N$ do
        if $j - i > MaxDist$ then
            $break$
        end if
        for $t = 1$ to $N_{w_i}$ do
            for $s = 1$ to $N_{w_j}$ do
                $weight \leftarrow Dependency(l^t_{w_i}, l^s_{w_j}, w_i, w_j)$
                if $weight > 0$ then
                    $AddEdge(G, l^t_{w_i}, l^s_{w_j}, weight)$
                end if
            end for
        end for
    end for
end for

**Score vertices in G**
for all $V_a \in Vertices(G)$ do
    $Score(V_a) \leftarrow Centrality(V_a)$
end for

**Label assignment**
for $i = 1$ to $N$ do
    $l_{w_i} \leftarrow argmax\{WP(l^t_{w_i}) | t = 1..N_{w_i}\}$
end for
```

First, a weighted graph of label dependencies is built by adding a vertex for each admissible label, and an edge for each pair of labels for which a dependency is identified. A maximum allowable distance can be set (MaxDist), indicating a constraint over the distance between words for which a label dependency is sought. Label dependencies are determined through the Dependency function, which encodes the relation between word senses. We conduct our evaluation using the following word similarity metrics: Leacock & Chodorow, Lesk, Wu & Palmer, Resnik, Lin, and Jiang & Conrath.

The Leacock & Chodorow [7] similarity is determined as:

$$Sim_{lch} = -\log \frac{length}{2 * D}$$

where length is the length of the shortest path between two concepts using node-counting, and D is the maximum depth of the taxonomy.

The Wu and Palmer similarity metric measures the depth of two given concepts in the WordNet taxonomy, and the depth of the least common subsumer (LCS), and combines these figures into a similarity score:

$$Sim_{wup} = \frac{2 * depth(LCS)}{depth(concept_1) + depth(concept_2)}$$

The measure introduced by **Resnik** returns the information content (IC) of the LCS of two concepts:

$$Sim_{res} = IC(LCS) \qquad\qquad IC(c) = -\log P(c)$$

and P(c) is the probability of encountering an instance of concept c in a large corpus.

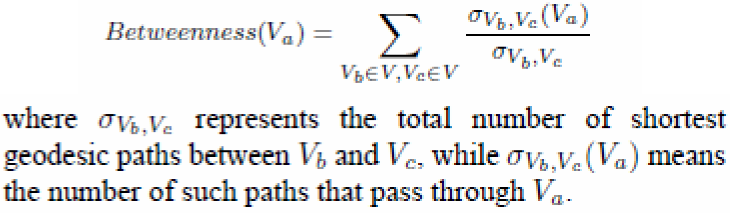
The next measure we use in our experiments is the metric introduced by Lin, which builds on Resnik's measure of similarity, and adds a normalization factor consisting of the information content of the two input concepts:

$$Sim_{lin} = \frac{2 * IC(LCS)}{IC(concept_1) + IC(concept_2)}$$

Finally, the last similarity metric considered is Jiang & Conrath:

$$Sim_{jnc} = \frac{1}{IC(concept_1) + IC(concept_2) - 2 * IC(LCS)}$$

Next, scores are assigned to vertices using a graph-based centrality algorithm. Finally, the most likely set of labels is determined by identifying for each word the label that has the highest Score. Note that all admissible labels corresponding to the words in the input sequence are assigned with a score, and thus the selection of two or more most likely labels for a word is also possible. We are using betweenness centrality algorithm. The **betweenness** of a node is defined in terms of how "in-between" a vertex is among the other vertices in the graph. Formally:



$$Betweenness(V_a) = \sum_{V_b \in V, V_c \in V} \frac{\sigma_{V_b,V_c}(V_a)}{\sigma_{V_b,V_c}}$$

where σ_{V_b,V_c} represents the total number of shortest geodesic paths between V_b and V_c, while $\sigma_{V_b,V_c}(V_a)$ means the number of such paths that pass through V_a.

Tools and Techniques Used:

1.	Language Used	Core Java
2.	IDE Used	NetBeans 7.4
3.	Algorithm Used	Graph Centrality And Dependency
4.	Dictionary Used	English WordNet2.1 Dictionary
5.	Operating System	Windows 7
6.	Processor	Intel i5
7.	Memory(RAM)	4GB
8.	Softwares	Notepad++
9.	Other Sources	Rita Wordnet Repository
10.	GUI Tools	Java Swing

Activity Chart Table :

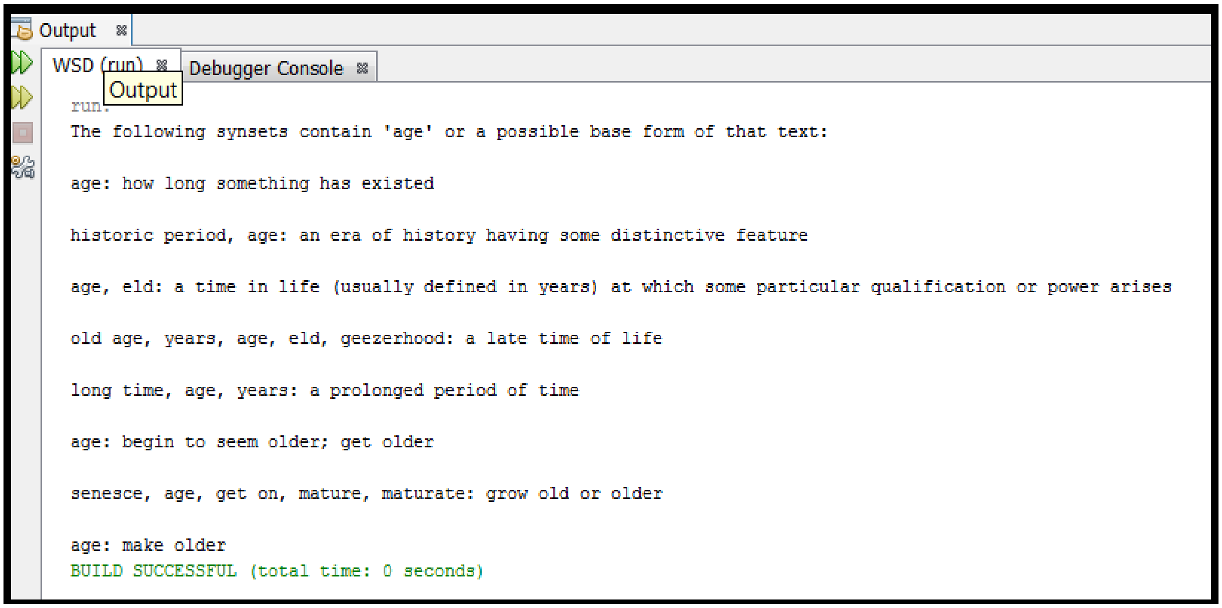
Activity	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5
Literature Survey	Jan 10th – Jan 25th	Feb 1st – Feb 15th		Feb 20th – Feb 25th	
Problem Formulation			Feb 16th – Feb 20th		
Learning Technique		Jan 25th – Jan 31th		March 15th – March 20th	
Design And Module (Coding)		March 22nd – March 30th	April 1st – April 10th		April 12th –April 20th
Interface Design	April 22nd –April 26th	May 3rd – May 5th			
Integrating Modules	May 5th – May 10th				

Report:

The software is developed which is implementing the proposed problem definition in a well-structured manner. It associates a word sense with a word in context using unsupervised approach.

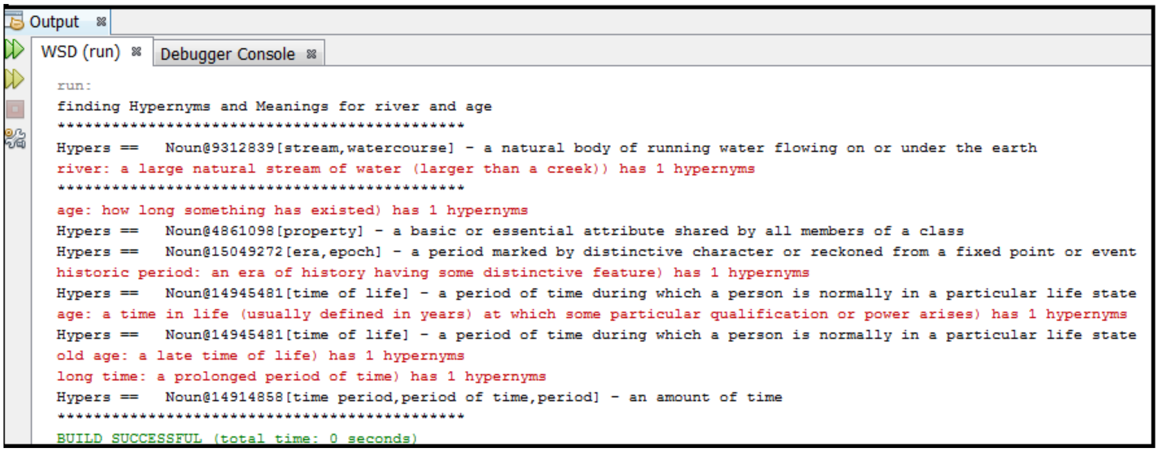
Working software snapshots:

Using wordnet in java for getting synsets:



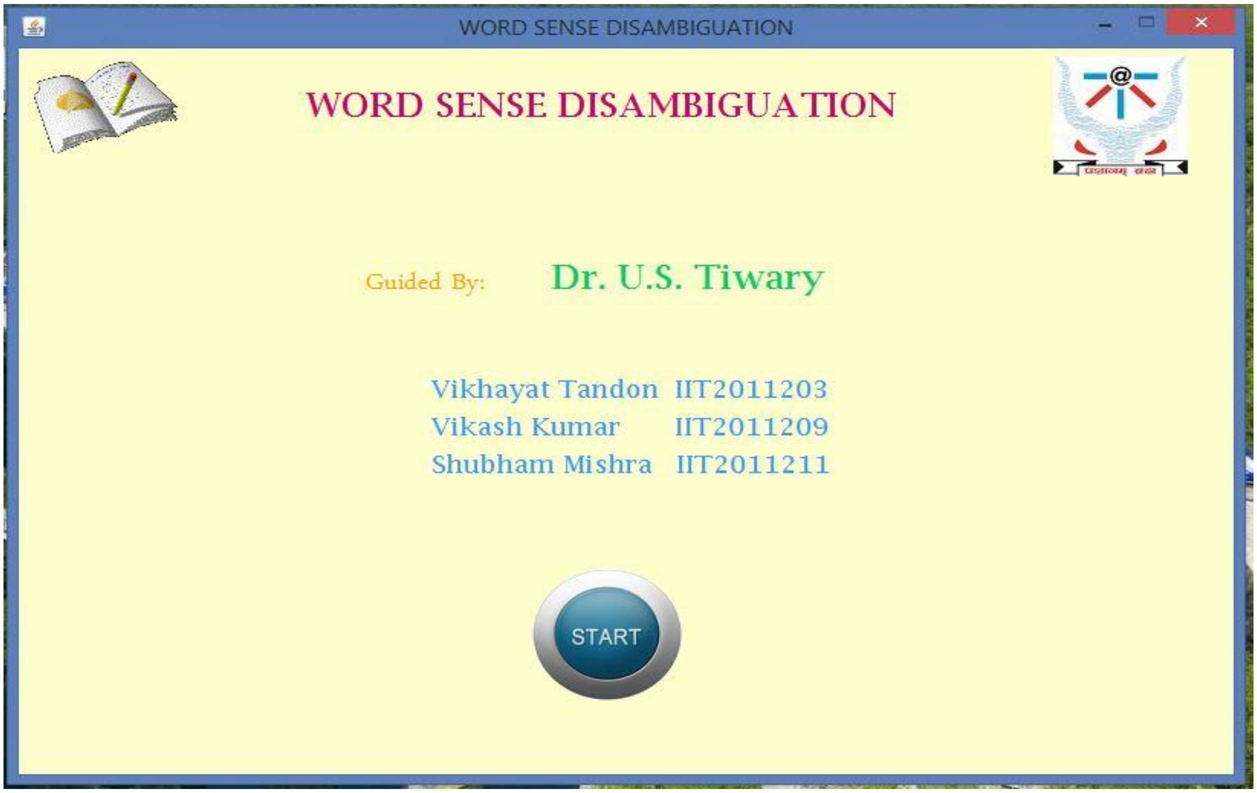
```
run:
The following synsets contain 'age' or a possible base form of that text:

age: how long something has existed

historic period, age: an era of history having some distinctive feature

age, eld: a time in life (usually defined in years) at which some particular qualification or power arises

old age, years, age, eld, geezerhood: a late time of life

long time, age, years: a prolonged period of time

age: begin to seem older; get older

senesce, age, get on, mature, maturate: grow old or older

age: make older
BUILD SUCCESSFUL (total time: 0 seconds)
```

Getting Hypernyms of words :



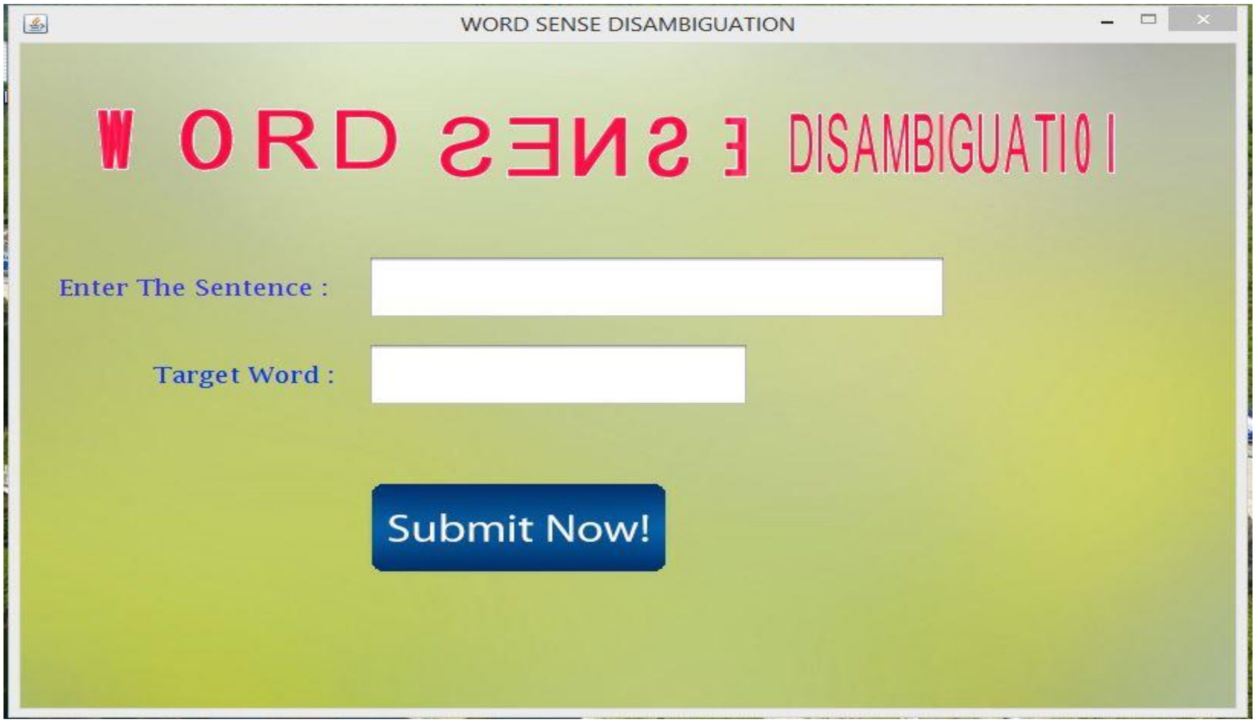
```
run:
finding Hypernyms and Meanings for river and age
*********************************************
Hypers ==   Noun@9312839[stream,watercourse] - a natural body of running water flowing on or under the earth
river: a large natural stream of water (larger than a creek)) has 1 hypernyms
*********************************************
age: how long something has existed) has 1 hypernyms
Hypers ==   Noun@4861098[property] - a basic or essential attribute shared by all members of a class
Hypers ==   Noun@15049272[era,epoch] - a period marked by distinctive character or reckoned from a fixed point or event
historic period: an era of history having some distinctive feature) has 1 hypernyms
Hypers ==   Noun@14945481[time of life] - a period of time during which a person is normally in a particular life state
age: a time in life (usually defined in years) at which some particular qualification or power arises) has 1 hypernyms
Hypers ==   Noun@14945481[time of life] - a period of time during which a person is normally in a particular life state
old age: a late time of life) has 1 hypernyms
long time: a prolonged period of time) has 1 hypernyms
Hypers ==   Noun@14914858[time period,period of time,period] - an amount of time
*********************************************
BUILD SUCCESSFUL (total time: 0 seconds)
```

Homepage:

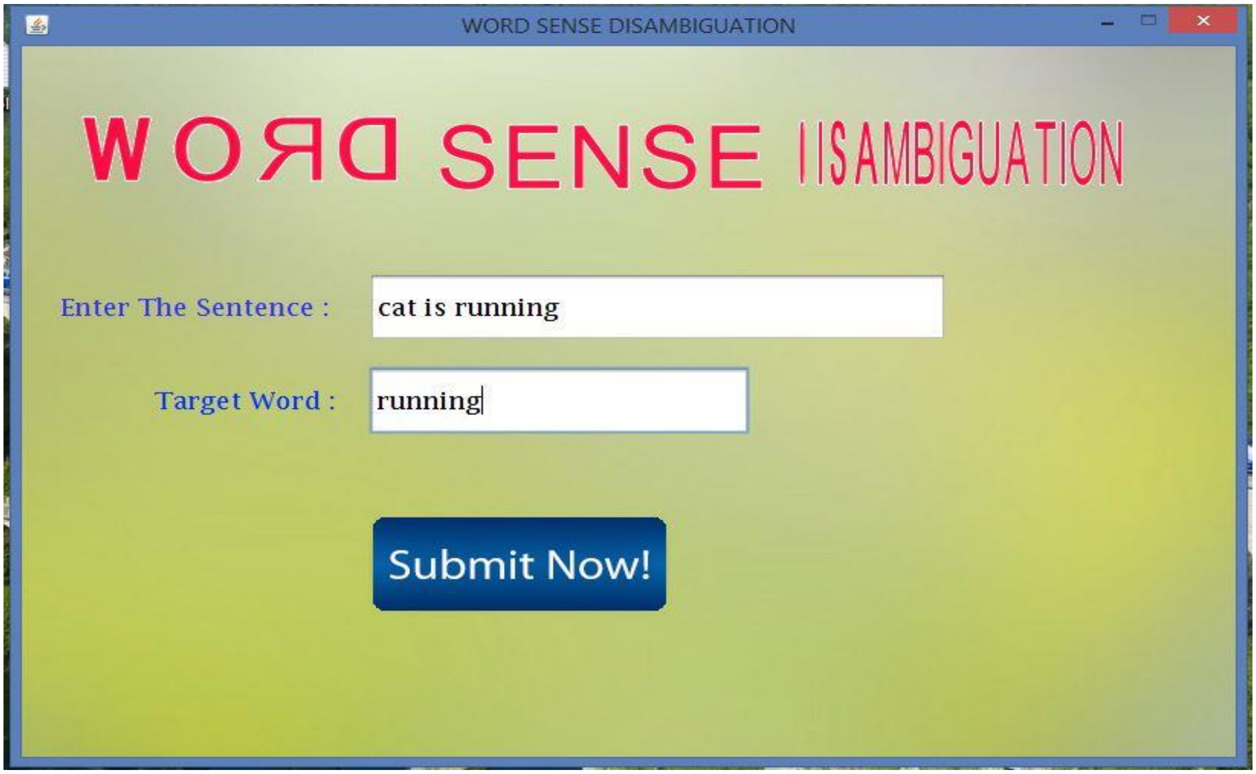


Guided By: Dr. U.S. Tiwary

Vikhayat Tandon	IIT2011203
Vikash Kumar	IIT2011209
Shubham Mishra	IIT2011211



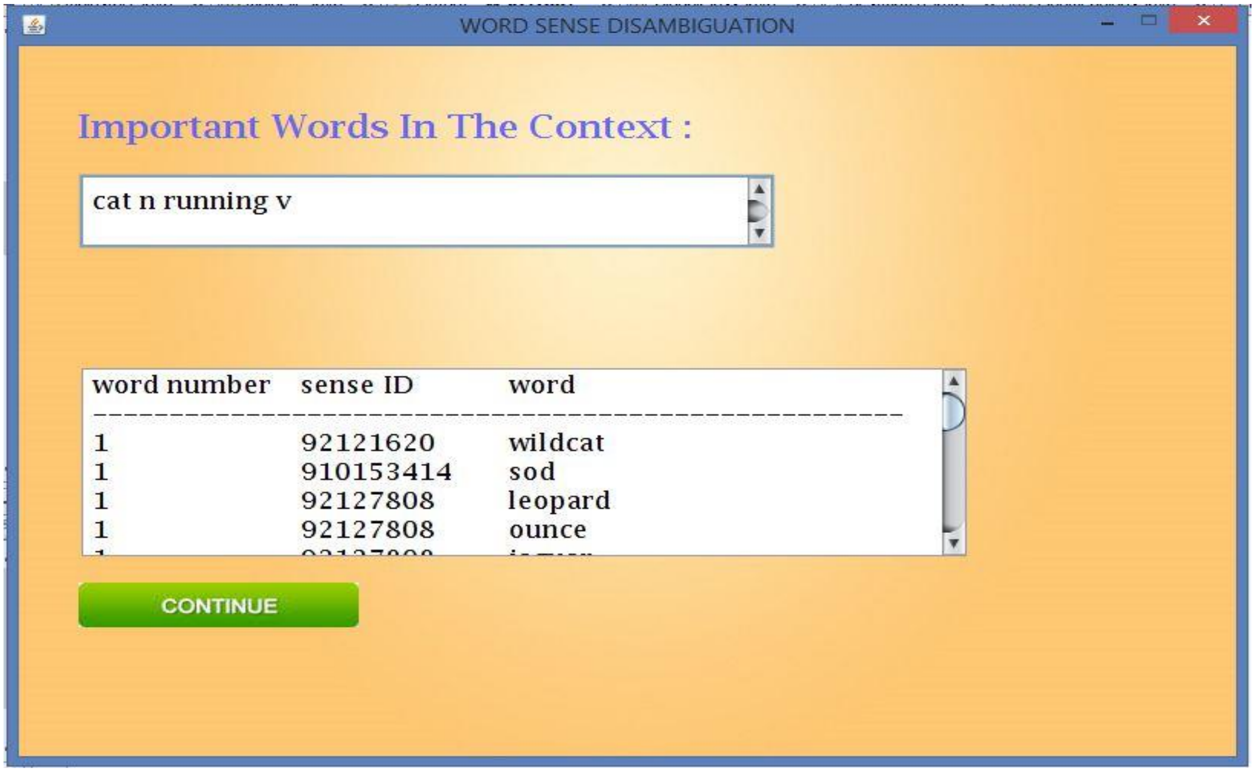
A simple example :



Enter The Sentence : cat is running

Target Word : running

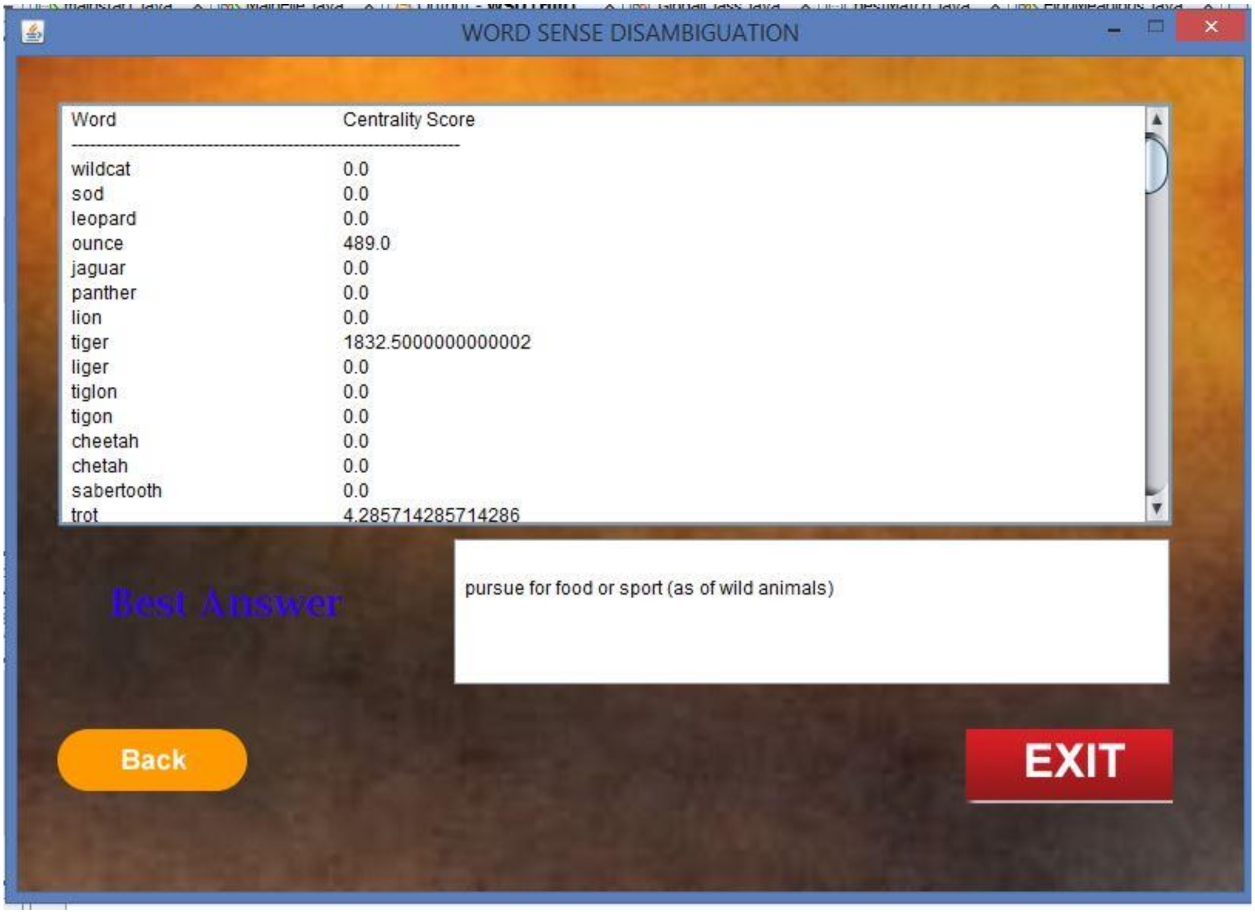
Important Words:



Important Words In The Context :

cat n running v

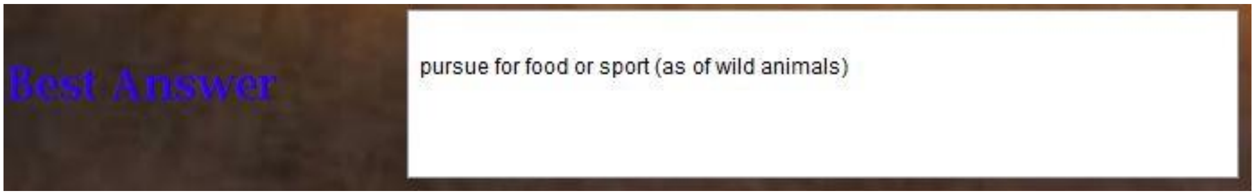
```
word number   sense ID        word
------------------------------------------------------
1             92121620        wildcat
1             910153414       sod
1             92127808        leopard
1             92127808        ounce
```



```
Word                          Centrality Score
---------------------------------------------
wildcat                       0.0
sod                           0.0
leopard                       0.0
ounce                         489.0
jaguar                        0.0
panther                       0.0
lion                          0.0
tiger                         1832.5000000000002
liger                         0.0
tiglon                        0.0
tigon                         0.0
cheetah                       0.0
chetah                        0.0
sabertooth                    0.0
trot                          4.285714285714286
```

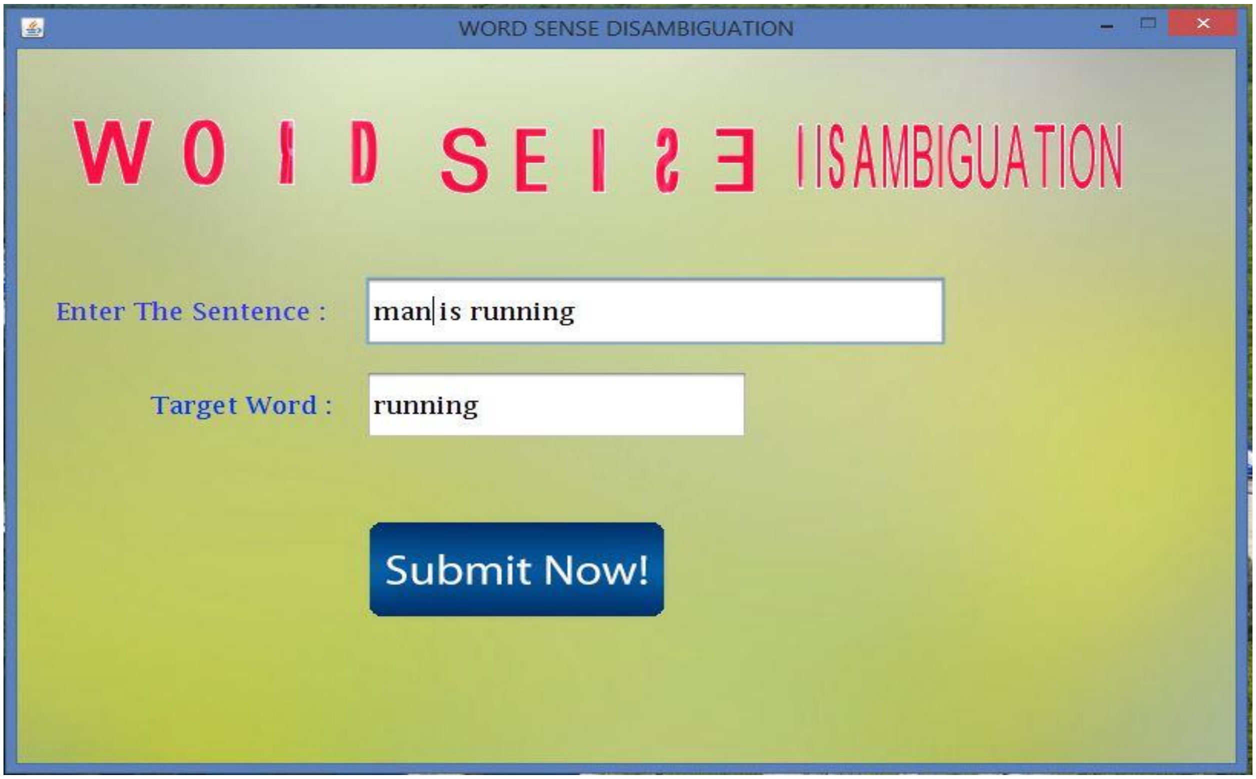
```
pursue for food or sport (as of wild animals)
```

Answer:



```
pursue for food or sport (as of wild animals)
```

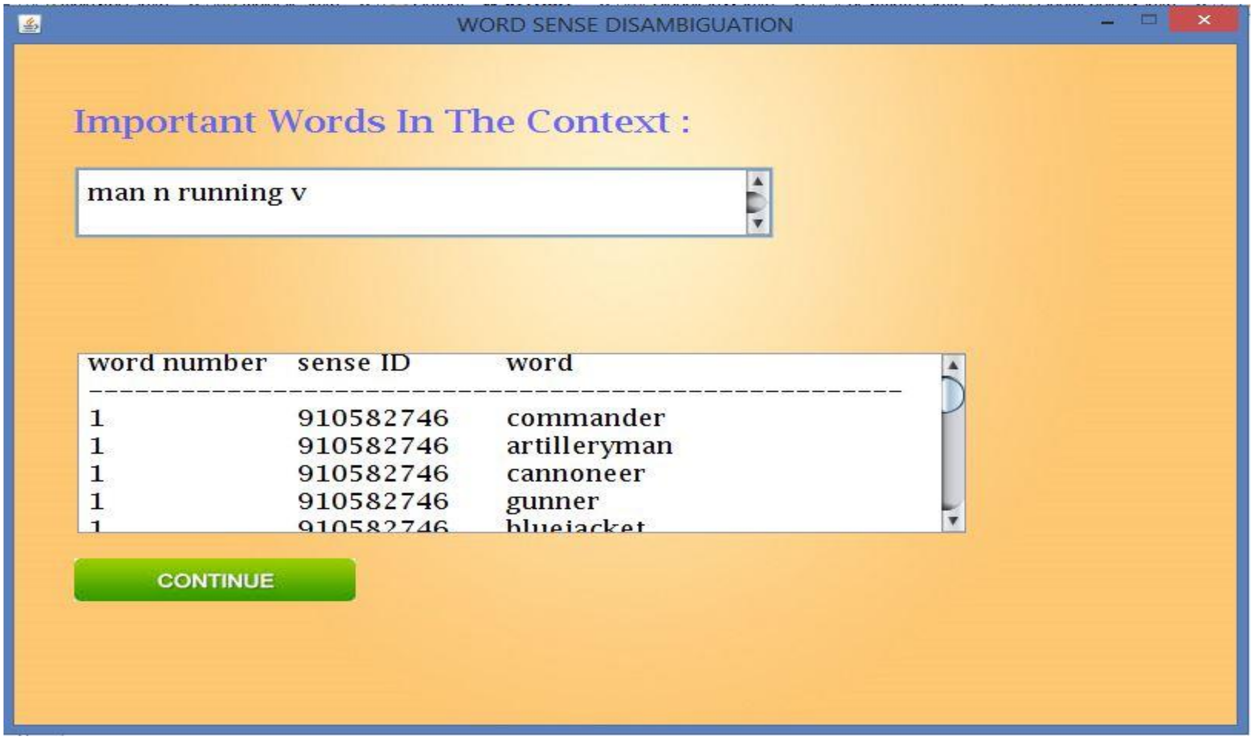
Other Example:



Enter The Sentence : man is running

Target Word : running

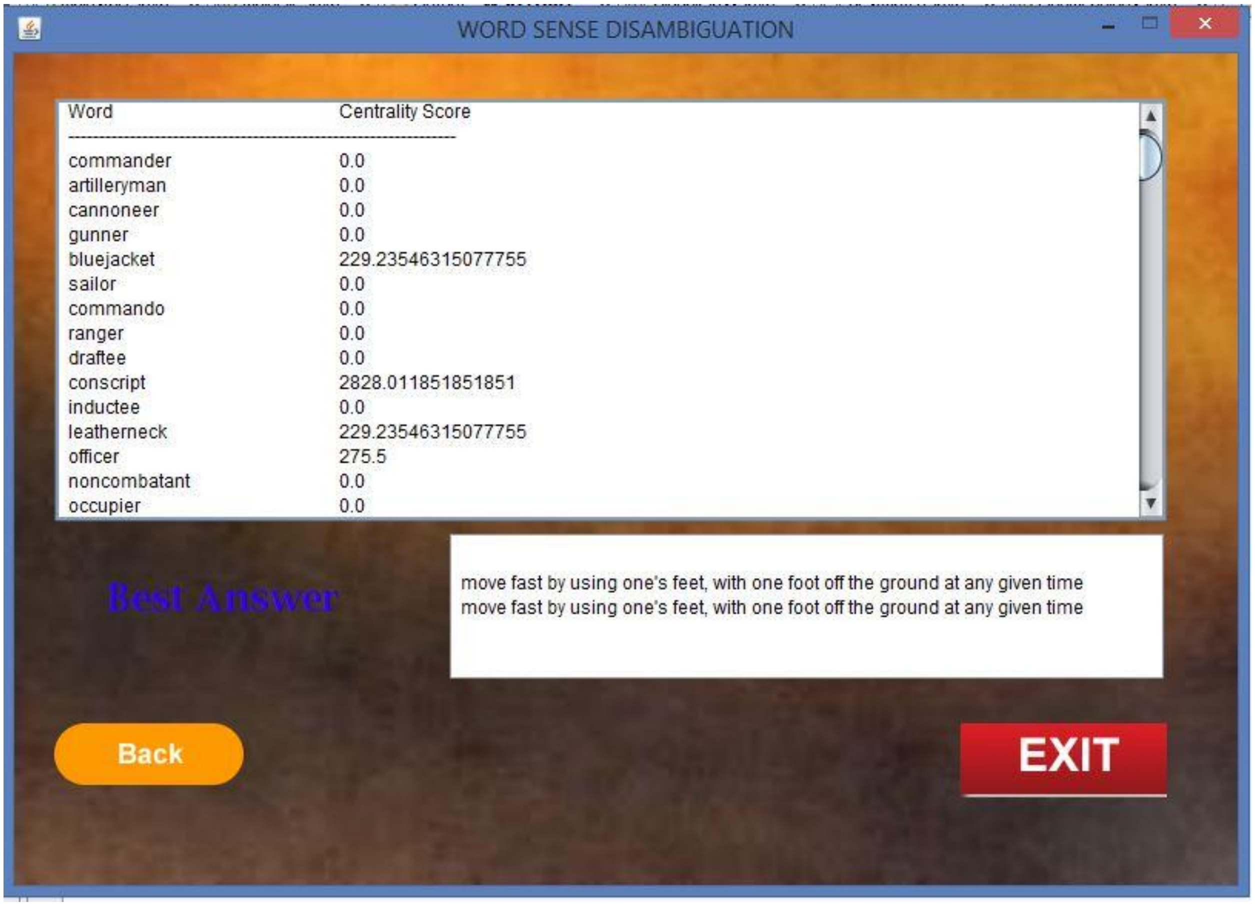
Important words:



Important Words In The Context :

```
man n running v
```

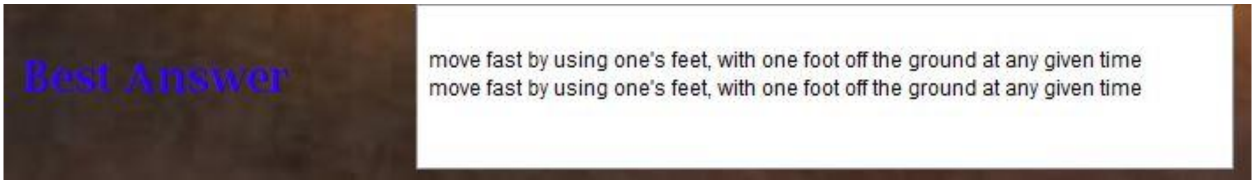
```
word number   sense ID       word
-------------------------------------------------------
1             910582746      commander
1             910582746      artilleryman
1             910582746      cannoneer
1             910582746      gunner
1             910582746      bluejacket
```



```
Word                                    Centrality Score
-----------------------------------------------------------
commander                               0.0
artilleryman                            0.0
cannoneer                               0.0
gunner                                  0.0
bluejacket                              229.23546315077755
sailor                                  0.0
commando                                0.0
ranger                                  0.0
draftee                                 0.0
conscript                               2828.011851851851
inductee                                0.0
leatherneck                             229.23546315077755
officer                                 275.5
noncombatant                            0.0
occupier                                0.0
```

```
move fast by using one's feet, with one foot off the ground at any given time
move fast by using one's feet, with one foot off the ground at any given time
```

Answer:



```
move fast by using one's feet, with one foot off the ground at any given time
move fast by using one's feet, with one foot off the ground at any given time
```

Consider two simple examples shown above in demo pics:

1. Cat is running
2. Man is running

Target word = “running”

For the first sentence the software returns the meaning of the target word as “chasing by animals for food” whereas in the second sentence it indicates the “physical activity”.

We used many such examples for further testing purpose and the results were quite satisfactory.

Conclusion

Our software provides a solution for unsupervised word sense disambiguation problem that is fast, scalable, efficient in terms of performance and accuracy and more useful for various applications such as Information retrieval, Machine Translation, etc. In Machine translations, it helps in better understanding of source language and generation of sentences in target language. It is also useful in Text to Speech translation i.e, when words are pronounced in more than one way depending on their meaning. For example: “lead” can be “in front of” or “type of metal”.

References:

[1] Ravi Sinha. Unsupervised Graph-based Word Sense Disambiguation, University of North Texas, June 2007

[2] Satanjeev Banerjee.Word Sense Disambiguation to WordNet, December 2002

[3] Michael Lesk. Automatic Sense Disambiguation, Bell Communications Research Morristown, 2004

[4] David Yarowsky. Unsupervised Word Sense Disambiguation, University of Philadelphia,1999

[5] Graph based centrality algorithm. http://en.wikipedia.org/wiki/Centrality

[6] M.Rada.The role of non-ambiguous words in natural language disambiguation. 2003.

[7] R. Mihalcea and D. Moldovan. A method for word sense disambiguation of unrestricted text, 1999.

Suggestion from Board members: