# UNL-ization of Numbers and Ordinals in Punjabi with IAN

Vaibhav Agarwal[1] and Parteek Kumar[2]

[1]M.E (S.E), Thapar University, Patiala, Punjab, India
vaibhavagg123@gmail.com
[2]Assistant Professor, CSED, Thapar University, Patiala, Punjab, India
parteek.bhatia@gmail.com

## ABSTRACT

*In the field of Natural Language Processing, Universal Networking Language (UNL) has been an area of immense interest among researchers during last couple of years. Universal Networking Language (UNL) is an artificial Language used for representing information in a natural-language-independent format. This paper presents UNL-ization of Punjabi sentences with the help of different examples, containing numbers and ordinals written in words, using IAN (Interactive Analyzer) tool. In UNL approach, UNL-ization is a process of converting natural language resource to UNL and NL-ization, is a process of generating a natural language resource out of a UNL graph. IAN processes input sentences with the help of TRules and Dictionary entries. The proposed system performs the UNL-ization of up to fourteen digit number and ordinals, written in words in Punjabi language, with the help of 104 dictionary entries and 67 TRules. The system is tested on a sample of 150 random Punjabi Numbers and Ordinals, written in words, and its F-Measure comes out to be 1.000 (on a scale of 0 to 1).*

## KEYWORDS

*UNL, IAN, UNL-ization, Punjabi NLP*

## 1. INTRODUCTION

The Universal Networking Language (UNL) is a non-speaking artificial language for representing, describing, summarizing, and storing information in a natural-language-independent format. In UNL approach, there are two basic different movements *viz.* UNL-ization and NL-ization. UNL-ization is the process of mapping / representing / analysing of the information conveyed by natural language utterances into UNL; NL-ization, conversely, is the process of realizing/manifesting/generating a natural language document out of a UNL graph. Both processes are totally independent. The UNL representation has been an interpretation rather than a translation of a given text. UNL has been exploited for several other different tasks in natural language processing, such as summarization, multilingual document generation, text simplification, semantic reasoning, and information retrieval. Sérasset and Boitet have viewed UNL as the future 'html of the linguistic content' [1]. Multilingual information processing through UNL had been proposed by Dave and Bhattacharyya *et al.* [2]. Currently, the main goal of UNL-ization process has been to convert the given information text into the intermediate language, *i.e.,* UNL. UNL represents knowledge with use of relations, attributes and universal words. Universal Words form vocabulary of UNL. These words correspond to nodes that are interlinked by relations or modified by attributes in a UNL graph. The concepts of UWs are divided into four categories, namely, nominal concepts, adjective concepts, verbal concepts, and adverbial concepts. A UW is a character string (an

DOI : 10.5121/ijnlc.2013.2307

English language word) followed by a list of constraints. Attributes are annotations made to nodes of UNL graph. Attributes denote the circumstances under which these nodes (or hypernodes) are used. In the UNL framework, relations describe semantic functions between two UWs. The set of relations is defined in the UNL specifications. The relations between the two nodes is of the form: *rel_name(node1,+att1;node2,+att2)*; here *rel_name* is the name of the relation , *node1* and *node2* are the two nodes between which the relations hold and *att1* , *att2* are attributes which are added to *nodes1* and *nodes2*, respectively.

## 2. Related Work

Earlier Enconversion was performed with the help of *'EnCo'* tool provided by the UNDL foundation to accomplish the task [3]. Martins *et al.* (1997) have proposed a prototype system for converting Brazilian Portuguese into UNL and DeConverting UNL expressions into Brazilian Portuguese with *'EnCo'* and *'DeCo'* tools, respectively [4]. Their system consists of three important sub-modules, namely, the lexical, the syntactic and the semantic modules. Martins *et al.* (2005) have noted that the *'EnCo'* and Universal Parser tools provided by UNDL foundation require inputs from a human expert who is seldom available and as such their performance is not quite adequate [5]. They have proposed the *'HERMETO'* system which converts English and Brazilian Portuguese into UNL. This system has an interface with debugging and editing facilities along with its high level syntactic and semantic grammar that make it more user friendly. Mohanty *et al.* (2005) have used Semantically Relatable Sequence (SRS) based approach for developing a UNL based MT system [6]. They have analyzed the source language using semantic graphs and used these graphs to generate target language text. Enconversion system for converting Punjabi language to UNL have been developed by Kumar and Sharma [7]. Dey and Bhattacharyya have presented the computational analysis of complex case structure of Bengali for a UNL based MT System [8]. They provided the details of the rule theory of *'EnCo'* and *'DeCo'* tools which are driven by analysis rules and generation rules respectively for Bengali language. These rules are based on case structure of '*kaaraks'* used in the Bengali language. Earlier the encoding tool used by UNDL foundation was '*EnCo'*. '*EnCo*' was a desktop application and the rules were categorized into five types, namely, left composition rules (+), right composition rules (-), left modification rules (<), right modification rules (>) and attribute changing rules (:) [7]. These rules were very complex. Graphical user interface of '*EnCo'* was not very user friendly. Now, UNL-ization is done by using IAN tool, in which node concatenation, node deletion and all other node operations can be done with the help of simple syntax. In IAN various resources like dictionaries, rulesets and corpus can be shared with other users. Now, since IAN is a web based application, IAN of every language can be integrated on centralised server and thus, interconversion and other benefits like information extraction can be availed irrespective of any language and geographic barrier.

## 3. IAN Framework

IAN performs three movements over the input file that are segmentation, *i.e.*, the division of input document into a series of processing units (sentences), that are processed one at a time; tokenization, *i.e.*, identification of the tokens (lexical items) of each sentence of input document; and transformation, *i.e.*, application of transformation rules of the grammar over each tokenized sentence in order to represent it as a UNL graph. IAN has six tabs; these are, Welcome, NL Input, Dictionaries, T-Rules, D-Rules, and IAN console. In NL input tab user has to provide the natural language document to be UNL-ized. In Dictionaries tab, NL-UNL dictionary is to be provided by the user. The dictionary must be provided according to the UNL dictionary specifications. User may have several different dictionaries, and could be loaded to process the same corpus, but they must be loaded in the correct order (because the order of the entries in the dictionary matter for tokenization). In T-Rules tab user has to provide the NL-UNL transformation grammar *i.e.*, the grammar to be used to process the natural language input. In D-Rules tab user provides the NL-UNL disambiguation grammar. DRules are used to control the tokenization and improve the results of the transformation grammar. In IAN console tab user gets the results. The IAN console brings the list of sentences appearing in the NL input, which could be processed one at a time, or can be processed in a range.

## 4. Implementation

In this paper, UNL-ization of numbers and ordinals, written in words in Punjabi language, has been explained with the help of different examples. The features of Punjabi language regarding Numbers and Ordinals has been presented in next sub section. The subsequent sub sections explains the UNL-ization process with the help of different Example sentences.

### 4.1. Features of Punjabi Language Regarding Numbers and Ordinals

Any single digit and two digit number written in Punjabi language in the form of words cannot be generated with any TRule because they have no repeating pattern. For example, consider the Punjabi words ਵੀਹ *vīh 'twenty'*, ਇੱਕੀ *ikkī 'twentyone'*, ਅਠਾਈ *aṭhāī 'twentyeight'*, with their English equivalent as twenty, twenty one, twenty eight, respectively, have '*twenty*' as the repeating pattern. Therefore, with the help of TRules these can be converted to UNL, without having any dictionary entry, if written in English language. Therefore, entries for single digit and double digit numbers, written in words in Punjabi language, needs to be done in the dictionary and all other numbers up to fourteen digits can be generated with the help of those dictionary entries and sixty seven TRules.

### 4.2. UNL-ization of Numbers

UNL represents the numbers in figures, written in the form of words. The dictionary entry of every single digit and double digit number like ਇਕ *ik 'one'*, ਦੋ *dō 'two'*, ਚਾਰ *cār 'four'*, ਗਿਆਰਾਂ *giārāṃ 'eleven'* , ਪੰਦਰਾਂ *pandrāṃ 'fifteen' etc.* are made into the dictionary with their respective attributes as shown below in the examples. While using rule based approach the sequences of rule is very important because the rules are fired in sequence as per matching. For example TRule 29 will be fired before TRule 34 irrespective of the fact that the nodes matching the criteria of TRule 34 are before the nodes matching the criteria of TRule 29 in the given input string. The UNL-ization process for number has been illustrated with the help of a simple example sentence (1).

Example 1: ਇਕ ਸੌ ਬਾਈ ...(1)

*ik sau bāī*
*one hundred twentytwo*

After the tokenization of example sentence given in (1) with IAN tool, five lexical items are identified as shown in (2).

[ਇਕ]{}"1"(LEX=U,POS=CDN,DIGIT,NUM=SNG)< pan,0,0>;

[ਸੌ]{-1}"ਸੌ"(TEMP)<xxx,0,0>;

[ਬਾਈ]{}"22"(LEX=U,POS=CDN,DOUBLE,NUM=PLR)<pan,0,0>;

Two blank spaces are also identified as :-
[]{}" "(BLK)<pan,0,0>; ...(2)

Here, *LEX* represents lexical category, *U* represents numeral, *POS* represents part-of-speech, *CDN* represents cardinal, *NUM* represents number whose value could be either *SNG* for singular or *PLR* for plural, *BLK* is the attribute given to the blank space, *TEMP* represents temporary entry, '*DIGIT*' indicates single digit number like ਇਕ *ik 'one'*, ਦੋ *dō 'two'*, ਚਾਰ *cār 'four' etc.*

*w*hile '*DOUBLE*' indicates two digit number like ਗਿਆਰਾਂ *giārāṃ 'eleven'*, ਪੰਦਰਾਂ *pandrāṃ 'fifteen' etc*. In <pan,0,0> 'pan' refers to the three-character language code for Punjabi according to ISO 639-3. First '0' represents the frequency of Natural Language Word (NLW) in natural texts. It can range from0 (less frequent) to 255 (most frequent). The second '0' refers to the priority of the NLW, used in case of NL-ization. It can range from 0 to 255. The process of UNL-ization of example sentence (1) has been illustrated in Table 1. Here, transliteration of each Punjabi word is shown only in Action Taken column.

Table 1. UNL-ization process for example sentence (1)

| Sno | TRule fired | Description | Action Taken |
|---|---|---|---|
| 1 | (%a,BLK):= ; | Here, *%a* refers to blank node [] having attribute 'BLK'. This rule is fired twice consecutively and it removes all the blank spaces. | Original nodes : [ਇਕ][][ਸੌ][][ਬਾਈ] *[ik][][sau][][bāī] [one][][hundred][][twentytwo]* Resultant nodes: [ਇਕ][ਸੌ][ਬਾਈ] *[ik][sau][bāī] [one][hundred][twentytwo]* |
| 2 | ({DIGIT\|DOUBLE},%a)("ਸੌ")(%b,{DIGIT\|DOUBLE}):=(%a,+HUNDRED)(%b); | Here, node [ਸੌ] *[sau] [hundred]* has been deleted and attribute HUNDRED has been given to previous node [ਇਕ] *[ik] [one]*. | Original nodes : [ਇਕ][ਸੌ][ਬਾਈ] *[ik][sau][bāī] [one][hundred][twentytwo]* Resultant nodes : [ਇਕ][ਬਾਈ] *[ik][bāī] [one][twentytwo]* |
| 3 | ({SHEAD\|CHEAD\|^BLK},%a)(DIGIT,HUNDRED,%x)(DOUBLE,%y):=(%a,ATT9)(%x &%y,-HUNDRED, -DIGIT,-DOUBLE,+000); | Here, SHEAD and CHEAD means sentence head and scope head, respectively and '^' is used as negation means logical NOT. This rule concatenates nodes [ਇਕ] *[ik] [one]* and [ਬਾਈ] *[bāī] [twentytwo]* to form a single node [ਇਕਬਾਈ] *[ikbāī][onetwentytwo]* as shown in the action taken column. All attributes are removed from final node and attribute 000 is added to it, to indicate, that it is a three digit number. | Original nodes: [ਇਕ][ਬਾਈ] *[ik][bāī] [one][twentytwo]* Resultant nodes: [ਇਕਬਾਈ] *[ikbāī] [onetwentytwo]* Now, ਇਕ *ik* '*one*' and ਬਾਈ *bāī*' '*twentytwo*' are replaced by their universal words '1' and '22', respectively, and the final output 122 is generated by IAN as shown in (3). |

The UNL generated is given in (3).
{org}
ਇਕ ਸੌ ਬਾਈ
{/org}
{unl}
[w]122:08[/w]
{/unl} ...(3)

Here, ':08' is the scope internally generated by the IAN tool. UNL-ization of numbers has been explained with the help of another complex example sentence (4).
Example 2: ਦੋ ਕਰੋੜ ਵੀਹ ਲੱਖ ਇੱਕੀ ...(4)

*dō karōṛ vīh lakkh ikkī*
*two crore twenty lakh twentyone*

After tokenization of example sentence given in (4) with IAN tool nine lexical items are identified as given in (5).

[ਦੋ]{-1}"2"(LEX=U,POS=CDN,NUM=PLR,DIG IT)<pa n,0,0>;

[ਕਰੋੜ]{-1}"ਕਰੋੜ"(TEMP)<xxx,0,0>;

[ਵੀਹ]{}"20"(LEX=U,POS=CDN,DOUBLE,NUM=PLR)<pan,0,0>;

[ਲੱਖ]{-1}"ਲੱਖ"(TEMP)<xxx,0,0>;

[ਇੱਕੀ]{}"21"(LEX=U,POS=CDN,DOUBLE,NUM=PLR)<pan,0,0>;

Four blank spaces are also identified as :-
[]{}""(BLK)<pan,0,0>; ...(5)
The process of UNL-ization of example sentence (4) is given in Table 2.

Table 2. UNL-ization process for example sentence (4)

| Sno | TRule fired | Description | Action Taken |
|---|---|---|---|
| 1 | (%a,BLK):=; | This Rule is fired four times to delete all four blank nodes as shown in Action taken column. | Original nodes :<br>[ਦੋ][][ਕਰੋੜ][][ਵੀਹ][][ਲੱਖ][][ਇੱਕੀ]<br>*[dō][][karōṛ][][vīh][][lakkh][][ikkī ]*<br>*[two][][crore][][twenty][][lakh][][t wentyone]*<br><br>Resultant nodes :<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ][ਲੱਖ][ਇੱਕੀ]<br>*[dō][karōṛ][vīh][lakkh][ikkī]*<br>*[two][crore][twenty][lakh][twentyo ne]* |
| 2 | ({DIGIT\|DOUBLE\|TRIPLE},%a)("ਲੱਖ")(%b,{DIGIT\|DOUBLE\|000\|0000\|00000}):=(%a,+LAKH)(%b); | Here, *%a* refers to node [ਵੀਹ] *[vīh][twenty]* having attribute DOUBLE and *%b* refers to node [ਇੱਕੀ] *[ikkī][twe ntyone]* having attribute DOUBLE. This rule deletes node [ਲੱਖ] *[lakkh] [lakh]* and gives attribute LAKH to the node *%a*. | Original nodes :<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ]**[ਲੱਖ]**[ਇੱਕੀ]<br>*[dō][karōṛ][vīh][lakkh][ikkī]*<br>*[two][crore][twenty][lakh][twentyo ne]*<br><br>Resultant nodes:<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ][ਇੱਕੀ]<br>*[dō][karōṛ][vīh][ikkī]*<br>*[two][crore][twenty][twentyone]* |
| 3 | ({SHEAD\|CHEAD\|^BLK},%z)({DIGIT\|DOUBLE\|TRIPLE},LAKH,%x)({000\|DIGIT\|DOUBLE\|STA | Here, *%x* refers to node [ਵੀਹ] *[vīh][twenty]* having attribute DOUBLE and LAKH, and *%y* | Original nodes:<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ][ਇੱਕੀ]<br>*[dō][karōṛ][vīh][ikkī]*<br>*[two][crore][twenty][twentyone]* |

| | | | |
|---|---|---|---|
| | IL\|CTAIL},^TEMP1,^00000,%y):=(%z)(%x)([[00]],[00],"00",THOUSAND,DOUBLE,LAKH_THOUSAND,%q)(%y); | refers to node [ਇੱਕੀ] *[ikkī][twentyone]*. This rule adds a node [00] between *%x* and *%y*. It also attaches the attributes THOUSAND, DOUBLE, LAKH_THOUSAND to this newly created node. | Resultant nodes:<br>[ਦੋ][ਕਰੋੜ] [ਵੀਹ]**[00]**[ਇੱਕੀ]<br>*[dō][karōṛ][vīh][00][ikkī]*<br>*[two][crore][twenty][00][twentyone]* |
| 4 | ({SHEAD\|CHEAD\|^BLK},%z){DOUBLE\|LAKH_THOUSAND},THOUSAND,%x)({DIGIT\|DOUBLE\|STAIL\|CTAIL},%y):=(%z)(%x)([[0]],[0],"0",HUNDRED,DIGIT)(%y); | Here, *%x* refers to node [00] which is preceded by a non-blank node being referred as *%z* and *%y* refers to node [ਇੱਕੀ] *[ikkī] [twentyone]* having 'DOUBLE' attribute. A new node [0] is added between *%x* and *%y*. It also attaches attributes 'HUNDRED' and 'DIGIT'. | Original nodes:<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ][00][ਇੱਕੀ]<br>*[dō][karōṛ][vīh][00][ikkī]*<br>*[two][crore][twenty][00][twentyone]*<br><br>Resultant nodes:<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ][00]**[0]**[ਇੱਕੀ]<br>*[dō][karōṛ][vīh][00][0][ikkī]*<br>*[two][crore][twenty][00][0][twenty one]* |
| 5 | ({SHEAD\|CHEAD\|^BLK},%z)(DIGIT,HUNDRED,%x)(DOUBLE,%y):=(%z,ATT 1)(%x&%y,-HUNDRED,-DIGIT, -DOUBLE,+000) ; | Here, *%x* refers to node [0] and *%y* refers to the node [ਇੱਕੀ] *[ikkī] [twentyone]*. This rule concatenates *%x* and *%y* to form the node [0ਇੱਕੀ] *[0ikkī][0twentyone]*. The attribute '000' is given to it and existing attributes DIGIT, DOUBLE, and HUNDRED are removed. | Original nodes:<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ][00][0][ਇੱਕੀ]<br>*[dō][karōṛ][vīh][00][0][ikkī]*<br>*[two][crore][twenty][00][0][twenty one]*<br><br>Resultant nodes:<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ][00][0ਇੱਕੀ]<br>*[dō][karōṛ][vīh][00][0ikkī]*<br>*[two][crore][twenty][00][0twentyone]* |
| 6 | (LAKH_THOUSAND,THOUSAND,%x)(000,%y)({CTAIL\|STAIL},%z):=(%x&%y,-THOUSAND, -000,-DIGIT,DOUBLE,-LAKH_THOUSAND,+00000)(%z); | Here, nodes [00] and [0ਇੱਕੀ] *[0ikkī] [0twenty one]* are concatenated to form a new node [000ਇੱਕੀ] *[000ikkī] [000twentyone]*. This new node is given the attribute 00000. The attributes 000, THOUSAND, DIGIT, DOUBLE and LAKH_THOUSAND are removed from it. | Original nodes:<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ][00][0ਇੱਕੀ]<br>*[dō][karōṛ][vīh][00][0ikkī]*<br>*[two][crore][twenty][00][0twentyone]*<br><br>Resultant nodes :<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ][000ਇੱਕੀ]<br>*[dō][karōṛ][vīh][000ikkī]*<br>*[two][crore][twenty][000twentyone]* |
| 7 | (DOUBLE,LAKH,%x)(00000,%y)({CTAIL\|STAIL},%z):=(%x&%y,-LAKH,-DIGI T,-DOUBLE,-00000,+0000000)(%z); | This rule concatenates nodes [ਵੀਹ] *[vīh] [twenty]* and [000ਇੱਕੀ] *[000ikkī][000twentyone]* to form a single node which is a 7 digit number as indicated by adding the attribute '0000000'. The attributes LAKH, DIGIT, DOUBLE and 00000 are removed. | Original nodes:<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ][000ਇੱਕੀ]<br>*[dō][karōṛ][vīh][000ikkī]*<br>*[two][crore][twenty][000twentyone]*<br>Resultant nodes :<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ000ਇੱਕੀ]<br>*[dō][karōṛ][vīh000ikkī]*<br>*[two][crore][twenty000twentyone]* |
| 8 | ({DIGIT\|DOUBLE\|TRIPLE},%a)("ਕਰੋੜ | Here, *%a* refers to node [ਦੋ] *[dō] [two]*. This rule deletes | Original nodes:<br>[ਦੋ][ਕਰੋੜ][ਵੀਹ000ਇੱਕੀ] |

| | | | |
|---|---|---|---|
| | "):=(%a,+CRORE); | the node [ਕਰੋੜ] *[crore][karōr]* and gives the attribute CRORE to node *%a*. | *[dō][karōr][vīh000ikkī]*<br>*[two][crore][twenty000twentyone]*<br>Resultant nodes:<br>[ਦੋ][ਵੀਹ000ਇੱਕੀ]<br>*[dō][vīh000ikkī]*<br>*[two][twenty000twentyone]* |
| 9 | (DIGIT,CRORE,^ARAB_CRORE,%a)(0000000,%b):=(%a&%b,-00 00000,-CRORE,-DIGIT,+00000000); | This rule concatenates nodes [ਦੋ] *[dō][two]* and [ਵੀਹ000ਇੱਕੀ] *[vīh000ikkī] [twenty000twentyone]* and gives the attribute 00000000 to this concatenated node while removes all other attributes. | Original nodes:<br>[ਦੋ][ਵੀਹ000ਇੱਕੀ]<br>*[dō][vīh000ikkī]*<br>*[two][twenty000twentyone]*<br>Resultant nodes:<br>[ਦੋਵੀਹ000ਇੱਕੀ]<br>*[dōvīh000ikkī]*<br>*[twotwenty000twentyone]*<br>Now ‘ਦੋ’ ‘*dō*’ ‘*two*’, ‘ਵੀਹ’ ‘*vīh*’ ‘*twenty*’ and ‘ਇੱਕੀ’ ‘*ikkī*’ ‘*twentyone*’ are replaced by their universal words and the final output 22000021 is generated by IAN as shown in (6). |

The UNL generated is given in (6).
{org}
ਦੋ ਕਰੋੜ ਵੀਹ ਲੱਖ ਇੱਕੀ
{/org}
{unl}
[w] 22000021:0F [/w]
{/unl} ...(6)
Here ‘:0F’ is the scope internally generated by the IAN tool.

## 4.3. UNL-ization of Ordinals

Ordinals numbers in Punjabi language contains ਲਾਂ *lāṃ* / ਵਾਂ *vāṃ* / ਥਾ *thā* / ਜਾ *jā* as suffix as in ਪਹਿਲਾਂ *pahilāṃ* ‘*first*’, ਇੱਕੀਵਾਂ *ikkīvāṃ* ‘*twenty first*’, ਚੌਥਾ *cauthā* ‘*fourth*’, ਤੀਜਾ *tījā* ‘*third*’, respectively. In UNL ordinals are represented in figures and attribute “@ordinal” is given to it to retain its semantics. During UNL-ization of ordinals, all rules fired will be same, with the only difference that after deleting blank spaces, a new rule as given in (7) will be fired. This rule will add “@ordinal” attribute to the corresponding node having suffixes ਲਾਂ *lāṃ* / ਵਾਂ *vāṃ* / ਥਾ *thā* / ਜਾ *jā*.

TRule:-({DIGIT|DOUBLE|CDN},%x)("ਵਾਂ"|"ਲਾਂ"|"ਥਾ"|"ਜਾ"):=(%x,ORD,att=@ordinal); ...(7)

This rule gives an attribute “@ordinal” to node having any one of the attributes *DIGIT*, *DOUBLE*, or *CDN* (being referred as *%x*) preceding any one of the nodes [ਵਾਂ] / [ਲਾਂ] / [ਥਾ] / [ਜਾ].

The UNL-ization process for ordinals has been illustrated with the help of a simple example sentence (8).

Example 3: ਇਕ ਸੌ ਬਾਈਵਾਂ ...(8)
*ik sau bāīvāṃ*
*one hundred twenty second*

After tokenization of example sentence (8) with IAN tool, five lexical items are identified as already given in (2) and it produce an additional lexical item as shown in (9).

[ਵਾਂ]{-1}"ਵਾਂ"(TEMP)<xxx,0,0>; ...(9)

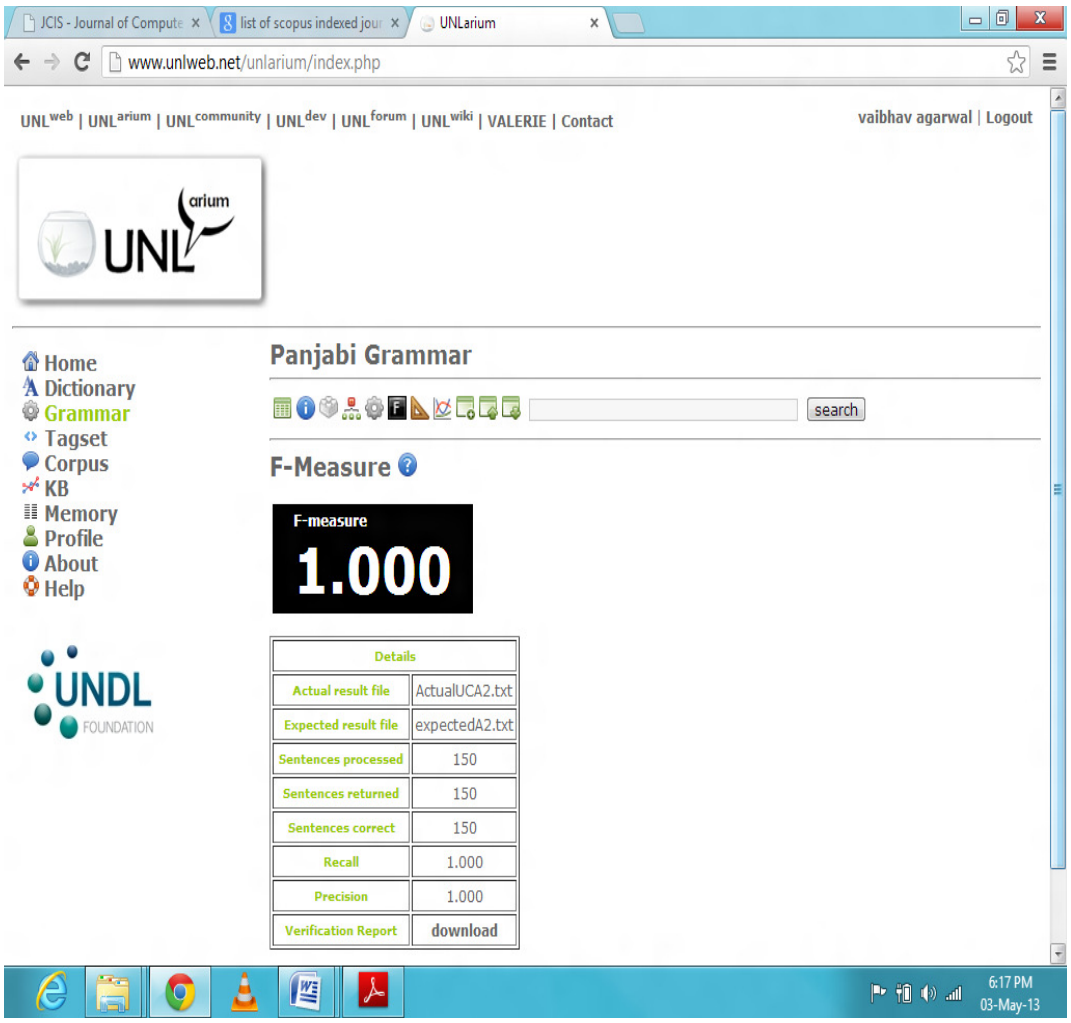
This lexical item help in firing of rule given in (7) to attach the attribute "@ordinal". The UNL of example sentence (8) is given in (10).

{org}
ਇਕ ਸੌ ਬਾਈਵਾਂ
{/org}
{unl}
[w] 122:09.@ordinal [/w]
{/unl} ...(10)

Here, ':09' is the scope internally generated by the IAN tool.

## 5. Results and Discussions

With the help of sixty seven TRules and one hundred and four dictionary entries all integer numbers and ordinals up to fourteen digits, written in words in Punjabi language, can be successfully UNL-ized using IAN tool. One hundred and fifty random sentences written in words, in Punjabi language, were successfully UNL-ized. Their F-Measure comes out to be 1.000 as shown in Figure 1. F-Measure is calculated with the help of online tool developed by UNDL foundation available at UNL-arium [9].


JCIS - Journal of Compute
list of scopus indexed jour
UNLarium
www.unlweb.net/unlarium/index.php
UNLweb | UNLarium | UNLcommunity | UNLdev | UNLforum | UNLwiki | VALERIE | Contact
vaibhav agarwal | Logout
UNL arium
Home
Dictionary
Grammar
Tagset
Corpus
KB
Memory
Profile
About
Help
UNDL FOUNDATION
search
6:17 PM
03-May-13

Panjabi Grammar

F-Measure

F-measure 1.000

| Details | |
|---|---|
| Actual result file | ActualUCA2.txt |
| Expected result file | expectedA2.txt |
| Sentences processed | 150 |
| Sentences returned | 150 |
| Sentences correct | 150 |
| Recall | 1.000 |
| Precision | 1.000 |
| Verification Report | download |

Figure 1. F-Measure of Punjabi Numbers and Ordinals

## 6. Conclusion and Future

In the field of Natural Language Processing, UNL has been viewed as the future 'html of the linguistic content' [1]. IAN tool is very effective for conversion of natural language sentences to UNL. It involves the process of UNL-ization with the help of NL-UNL dictionary and TRules. The proposed system is able to generate UNL for fourteen digit Punjabi numbers and ordinals written in words with the help of one hundred and four dictionary entries and sixty seven TRules.

**In future**, the work can be extended for converting numbers and ordinals of any size, floating point numbers, and mixed fractions to UNL. UNL-ization of all the major part-of-speech like nouns, adjectives, prepositions, determiners, conjunctions, verbs, *etc.* can further be carried out with IAN tool. Since UNL captures semantics of a given natural language resource, semantic based searching system can be developed.

## REFERENCES

[1] G. Sérasset, C. Boitet, C 2000, "UNL as the future "html of the linguistic content" & the reuse of existing NLP components in UNL-related applications with the example of a UNL French deconverter", *Proc. Int. Conf. on Computational Linguistics*, Saarbruecken, Germany, 2000, pp 768-774.

[2] S. Dave, J. Parikh, P. Bhattacharyya 2001, "Interlingua based English Hindi machine translation and language divergence", *J. of Machine Translation (JMT)*, volume 16:(4), 2001, pp 251-304.

[3] UNDL Foundation, "http://www.undl.org", December, 2012.

[4] R.T. Martins, L.H.M Rino, O.N. Osvaldo, R. Hasegawa, M.G.V. Nunes 1997, "Specification of the UNL-Portuguese enconverter-deconverter prototype", *São Carlos: ICMSC-USP*, 1997, pp 1-10.

[5] R.T. Martins, R. Hasegawa, M. Graças, V. Nunes 2005, "Hermeto: A NL–UNL Enconverting Environment", *Universal Network Language: Advances in Theory and Applications, Ed(s) Cardenas J, Gelbukh A, Tovar E,* México, Research on Computing Science: 2005, pp 254-260.

[6] R. Mohanty, A. Dutta, P. Bhattacharyya 2005, "Semantically relatable sets: building blocks for representing semantics", *Proc. 10th MT Summit, Phuket*, 17 May 2005, pp 1-8.

[7] P. Kumar, R.K. Sharma,"Punjabi to UNL EnConversion System", *Springer: Sadhna, Academy Proceedings in Engineering Sciences*, volume 37:(2), April 2012, pp 299–318.

[8] Dey, K., Bhattacharyya, P 2005, "Universal Networking Language based analysis and generation of Bengali case structure constructs" , *Universal Network Language: Advances in Theory and Applications, Ed(s) Cardenas J, Gelbukh A, Tovar E,* México, Research on Computing Science, 2005, pp 215-229.

[9] UNL-arium,"http://www.unlweb.net/unlarium/index.php?lang=pa", May 2013.

**Authors**

**Vaibhav Agarwal** was born in Bareilly, Uttar Pradesh, India, in 1988. He received his B.Tech degree in Computer Science engineering from U.I.E.T, Kurukshetra University, in 2010. Currently he is a postgraduate student of Software engineering from Thapar University. His interest is focused on Natural Language Processing.
**E-mail:** vaibhavagg123@gmail.com

**Parteek Kumar** is currently working as an Assistant Professor in Computer Science and Engineering Department, Thapar University. His specialization is in the fields of Machine Translation and Database Management system.
**E-mail:** parteek.bhatia@gmail.com