Urdu Morpology with Finite State Tranducers

University Tuebingen

ISCL

Prof: Kurt Eberle

Safeer Ahmed Mian

3968013

Abstract:

This paper introduce and discuss a methodology in which Urdu Language can be marked hence when someone wants to know the grammatical meaning of the sentence can extract the information out of the sentence. To accomplish this I have used Finite State Transducers. Finite State Transducers would save the information of the words dividing the word into stem and the rest part of the word would be used to to save the information of the gender along with which word class it belongs to. As for verbs in Urdu the verbs are used according with the gender single or gender prular we divide the verb also in stem to get the verb and rest part to get the infomation of the gender. In the end I would talk about how this method can be used in future.

Table of Contents

Introduction:

Urdu (اردو), it is the language which was originated in India. Later on it gets spread to the nearby continents and captured the name as Indo- European language. But, still the tradition stays in Hyderabad, Lucknow & Lahore. The script of the language is Arabic ie; the language of Qur'an. The developments of its dialects are similar to Hindi. Lahore and Karachi to Delhi and Calcutta captured the area to maintain the dialects of the language. This language is formally given the title as "Language of court and camp", which is given in Urdu as "Zaban-e-Urdu-e-Moalla (زبانِ اردوِ معلہ)". The word 'Urdu' was originated from Turkish, gives the meaning as "army". Urdu is the national language in Pakistan and gives twentieth place in the world.

Although Urdu is quite a popular language not much work have been done on this language. This paper discusses how we can analyze the Urdu language. With the idea that I propose we can code the Urdu language and make an analyzer out of it. For this purpose I have used Finite State Transducers with which I will show you how can we divide the sentence in words and can deduct the grammatical meaning of the sentence.

Distinguishing the genders:

In Urdu Language gender distinguishing is very important because the whole sentence is then structured according to the gender. Unlike in English, for example, If the subject is the a male then the verb of the male gender has to be used or it the subject a female then the female verb has to be used for the sentense. For example

لڑکا بھا گا

The boy ran

لڑکی بھاگی

The girl ran

THE MASCULINE GENDER: "JINS É MUZEKKER"

(a) All Male names:

The names in Urdu speaking nations have some meaning attach to it so all the names which have meaning referring to profession or have male quality are marked as male names irrespective of ending.

ex: Feroz , Shehzad , Umer , Ali, Elyas, Ahmed etc.

(b) Nouns ending with a consonant followed by an Alif (ا -a):

ex: Lerka (boy), Kutta (dog), Hira (Gem).

(c) Nouns ending with a "ha" (ہ) :

Bacha (child), Ghadha (donkey),Qissa (tale), Perdha (viel), Agrah

(d) Nouns ending with -an (nasalized N, as "n" in the French name "Chopin":

Ex: Pakistan, Firozan

(e) Nouns ending with "u" and "o" (Urdu :و)

Ex: Alu (potato), Bhalu (teddy bear), Jadu (magic) etc.

Important exceptions: Arzu (desire, need), khusbu (fragrance)

(f) Nouns ending with the suffix "pan" (Urdu:پن)

Pagelpan (state of insanity),Apnapan (feeling of belong), Becpan (childhood)

(g) Most nouns (borrowed from Arabic) which begin with the prefix "M-" of locality:

Ex: Mkan (House), Mqam (locality),
Important exceptions: Mesjid (mosque), Mehfil (gathering), Menzil منزل (destination) , Mejlis (The Parliament/ or meeting)

(h) Nouns ending with the suffix "-istan" (Urdu:ستان):

Ex: Gulistan (Garden), Hindustan (India), Registan (Desert)

THE FEMININE "JINS É MUENNIS" :

(i) All female names:

Shenaz , Ayesha, Maliha , Feryal.
AND ALL professions OR dispositions when referring to a female, IRRESPECTIVE of endings:
Vezir - e- Azem (Prime Minister), Hemshira (sister), Alima (Scholar), Rani (Queen) etc.

(j) All nouns ending in "i" and " ia" (Urdu : ی یا)   :

Ex: shadi (wedding), Lerki (girl), Kehani (story), Roti (bread), Ciriya   (Bird) etc.

(k) All nouns ending in "-t" or "-et" or "-at" (Urdu: ت)

Ex: Halet (condition), Behist (paradise), Jennet (heaven), Lanet (curse), shoret (fame), shanakhat (identity) etc.
Two important exceptions: Vakt (time), and shrbat (juice)

(l) All nouns ending in "-gi" (Urdu :گی)

Ex: Zindegi (Life), Bendegi (Slavehood)

(m) All nouns ending in "sh" (Urdu:ش)

Ex: Lerzish (trembling), Malish (massage), Verzsh (excercise), Danish (knowledge).

(n) Abstract knowns formed by dropping "-na" in verbs, such as in:

"Mar" from "Marna" (to hit) – Uski mar khana asann nehîn (to bear his beatings is not easy),
"Len-Den" from "Lena and Dena" (to take and to give) – Uski lenden kharab ha. (His measure is faulty.) etc.

Although these rules are there but in Urdu there are many exceptions for eg Fida can be both name for male or female. There are also words which have no rules of marking but you just have to know there gender kalam (pen) is fem., "ghar" (house) "kitab" (book) is fem, "Miez" (table) is male etc

Lastly, usually the same rules, as above, apply to English words used in Urdu. "Country" is feminine as it ends with the "i" sound.

Verbs in Urdu:

The verbs in Urdu also have to have a gender marker. So in sentences the verbs are marked as follows:

All verbs ending with ا are male single

All verbs ending with ی are female single + femal plural

All verbs ending with ے are male plural

Tenses In Urdu:

In Urdu, like in English we have three tenses. Present, Past and Future. But as we saw above we have to use verbs with the respective genders. The tense marking in Urdu is very easy to distinguish but we have to keep in mind even the marking words used to describe which form of tense it falls into we have to keep in mind the gender. For example

لڑکی جا رہی تھی

The girls was going

لڑکا جا راہا تھا

The boy was going

In this sentense تھی is used for past tense female تھا for past tense male.

To explain the concept of tensese I will take one sentence of English and show you the tenses in Urdu. So oue English Sentence is.

I write a letter

so lets make all the tenses in urdu to understand.

میں خط لکھتا ہوں (male present)

میں خط لکھتی ہوں(female present)

میں خط لکھتا تھا(male past)

میں خط لکھتی تھی(female past)

میں خط لکھوں گا(male future)

میں خط لکھوں گی(female future)

میں خط لکھتا ہوں گا(male future continuous)

میں خط لکھتا ہوں گی(female future continuous)

میں خط لکه راھا ھوں(male present continuous)

میں خط لکه راھی ھوں(female present continuous)

میں خط لکه راھا تھا(male past continuous)

میں خط لکه راھی تھی(female past continuous)

میں خط لکه چکا ہوں(male present perfect)

میں خط لکه چکی ہوں(female present perfect)

(male past perfect)میں خط لکه چکا تھا

(female past perfect)میں خط لکه چکی تھی

(male future perfect) میں خط لکه چکا ہوں گا

(female future perfect)میں خط لکه چکی ہوں گی

Grammatical persons:

In Urdu, Like in English we have first person, second person, third person

First person :	میں
Second person :	آپ،تو
Third person :	وہ،اس

To implement Urdu language morphology I have used Finite state Transducers.

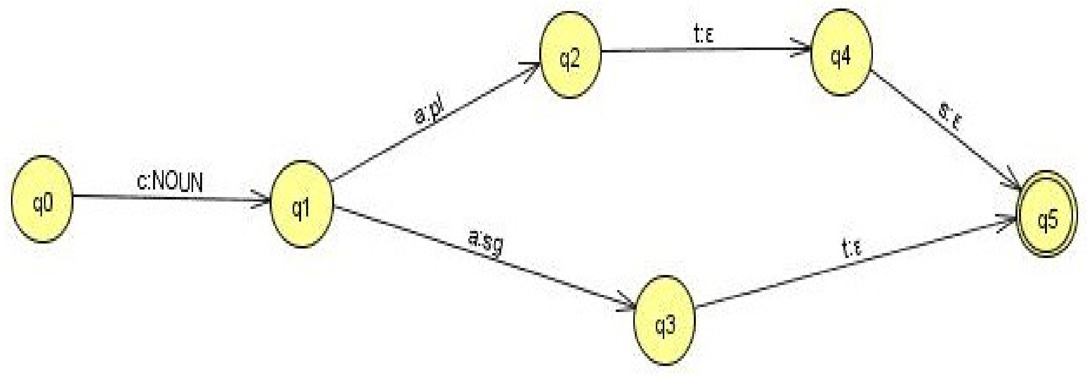
Finite State Transducers:

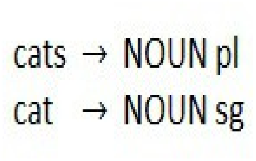
Finite State Transducers are almost similar to simple Finite State Automata but are a bit powerful. Finite State Transducers contains two memory tapes input tape and output tape. An FST will read a set of strings on the input tape and generates a set of relations on the output tape.

Finite State Transducer (FST) is a 7-tuple($Q, \Sigma, \Gamma, \delta, \omega, q0, F$)

1. Q is a finite set of states
2. Σ is a finite set input alphabet
3. Γ is a finite set of output alphabet
4. $\delta : Q \times \Sigma \rightarrow Q$ is the transition function
5. $\omega : Q \times \Sigma \rightarrow \Gamma$ is the output function
6. $q0 \in Q$ is the start state
7. $F \subseteq Q$ is the set of accept states

Example of a FST would be




cats → NOUN pl
cat → NOUN sg

ıd

Task:

For me the task is to implement a FST from the transition table which looks something like this for example:

Current State	Next State	Input	Output
q0	q1	c	NOUN
q1	q2	a	pl
q1	q3	a	sg
q2	q4	t	ε
q4	q5	s	ε
q3	q5	t	ε

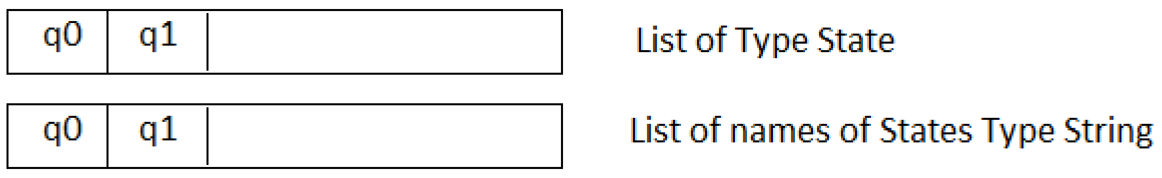
Final States:
q5

Implementation:

To implement Finite State Transducers I worked a lot with lists. So the classes involved are Transition State and FST

A transition would have next state input transition and output transition.
A state would have a name and list of transitions.
And Finite State Transducers would hvae a list of States.

To have an easy access of the states what we do is we creat two list in Finite State Transducers class. One List of state Type and one List of names of state of Type String.
Now when a new State is created it is added in the List of states and also in the List of names of states.



By doing so it helps us in finding the required state by looking at the index of the required state in the list of names list. For example if we want to find the index of istate and ostate we can now get this by

```
index1=allnames.indexOf(istate);
index2=allnames.indexOf(ostate);
```

allnames is the List<String> storing the names and istate and ostates are Strings. Now as discussed earlier we can get the same state from the List<State> by using these index

```
i=all.get(index1);
o=all.get(index2);
```

where i and o are now the input state and output states respectively.

Add Function:

In this implementation we also have the possibility of defining the cost of the transition. Our add function have the following defination

```
	void Add(String istate,String input,String ostate,Double cost,String
output)
```

In the add function we have 4 possibilities
1) input and output state does not exist.
2) input exists and output does not.
3) input does not exist but output does.
4) input and output both exists.

Checking by our required conditions we jump to that area of the section for example if we take a look at the first condition that is when both states are not

present we simply creat a new states and add them to all and allname lists and set the value of transtions. We also set a flag up to see if the input and output results are same or not.

```
	if(!allnames.contains(istate) && !allnames.contains(ostate))
			{
				State i=new State();
				State o=new State();
				i.isfinal=false;
				o.isfinal=false;
				Transition t=new
				Transition(input,ostate,cost,output,up);
				i.transi.add(value);
				i.transitions.add(t);
				i.name=istate;
				o.name=ostate;
				i.link.add(o);
				allnames.add(istate);
				allnames.add(ostate);
				all.add(i);
				all.add(o);
				size++;    //i added
				size++;    //o added
			}
```

The rest of the conditions are similar in thoes we find the index of the required states with the help of the other name list (allnames) and make the required changes.

Now we can have a senario in which a for example

```
q1    q2    a    a
        And
q1    q2    a    ab
```

To cater this problem what we do is we save the transition in the following pattern

```
q1     q2     a      a ab
```

UpFinal:

Now we have a function which take the input string and gives the output result. For we would have to divide the input string into the tokens. In Finite State Transducers there is possibility of ambiguity (more than one different derivations of same words). So we can have more than one output for this our function would have the following defination

```
List<String>  upfinal(String word)
```

The functionality of this method is to first split the words into different tags and since the FST will always have one initial state we would compare the tags with the transition list in the q0 state which is done by the following method

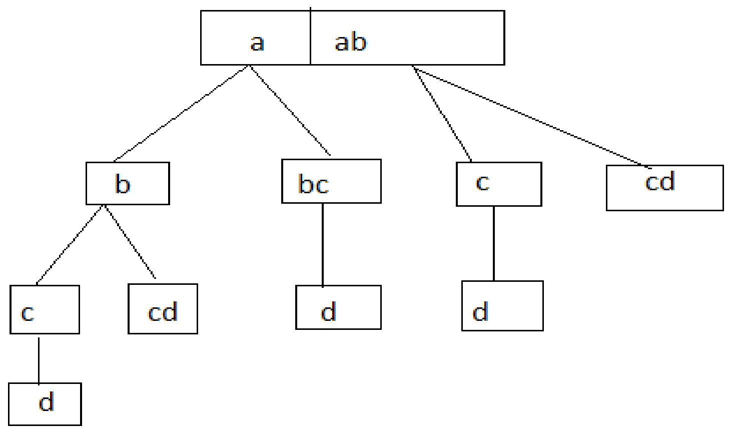
```
List<Transition> possible_Transitions(State q,String word)
```

which uses the method from State class

```
Transition min(String value)
```

To take care of what we are using as tag and what it converts into we use final_output which is input values and ch_output which is the change or output values. For example suppose we have the following transitions

```
            t.Add("q1","a", "q2", 0.4,"a");
            t.Add("q1","a", "q2", 0.4,"abc");
            t.Add("q1","a", "q2", 0.4,"d");
            t.Add("q2", "d", "q5", 0.0, "c");
            t.Add("q2", "b", "q3", 0.2,"b");
            t.Add("q2", "bc", "q4", 0.2,"bc");
            t.Add("q3", "c", "q4", 0.0, "c");
            t.Add("q3", "cd", "q4", 0.0, "cd");
            t.Add("q4", "d", "q5", 0.0, "d");
            t.Add("q4", "d", "q5", 0.0, "f");
            t.Add("q1","ab", "q3", 0.4,"ab");
initial,input,next state,cost,output
```



The image above is just to show how the derivation tree would be if the input and output are the same. This all is done by possible_transition and later on to update the our output we have a update_final method. Lets assume the first row we have two entries and each entry has other entries so we would have to do a tree traiversal method to continue doing so we have a Repete1 function which does this and does depth traiversal.

```
       List<String> Repe1(List<String> final_output,List<String>
output,List<Transition> transition,String word,List<String> ch_output)
```

This method returns us a List of Strings which are basiclly the path followed to achieve the given word. This path is a bit modified as we add all the output entries seperated with (£).

Having this done we have a final function named finally_donewhich merge all these entries in a single string entries into different paths

```
                    List <String> final_done(String words)
```

For example if we have the entry £a ab£b this function will return us a List containing two entries ab and abb.

The implementation also contains a exist function which works on the same principle.

Stemming of the words:

So based on the knowlodge we had gathered What i propose is to have a our own definition of the language by spliting the word in a unchanged and a changed form for example:

لڑکے لڑکیاں لڑکی لڑکا

boy,girl,girls,boys

In all of these words we see that all the words have the starting same and the ending are different so i would say in a FST we would have something like this:

لڑک :+Noun+

ا :+male+sg+

ی :+female+sg+

یاں:+female+pl+

ے:+male+pl+

Similarly for checking the tenses what I propose is to make

تھا تھی تھے تھیں

femal pl past,male pl past,female past,male sg past

ہے ہیں

male femal pl present,male female sg present

ہوگا ہوگی ہونگی ہونگے

male pl future,female pl future,female sg future,male sg future

چکی چکا چکے چکیں

femal pl perfect ,male pl perfect,male sg perfect,female perfec

راھے راھی راھا راھیں

femal pl continuous, male sg continuous,female continuous,male pl continuous

Even for verbs what I would suggest is we look the verb ending if it ends with ا or ے it is male or male plural but with ی it can be either female single or female plural so it it is hard to decide if it is single or plural and we would have to read the sentence in full and extract the context and then have exact marking for example

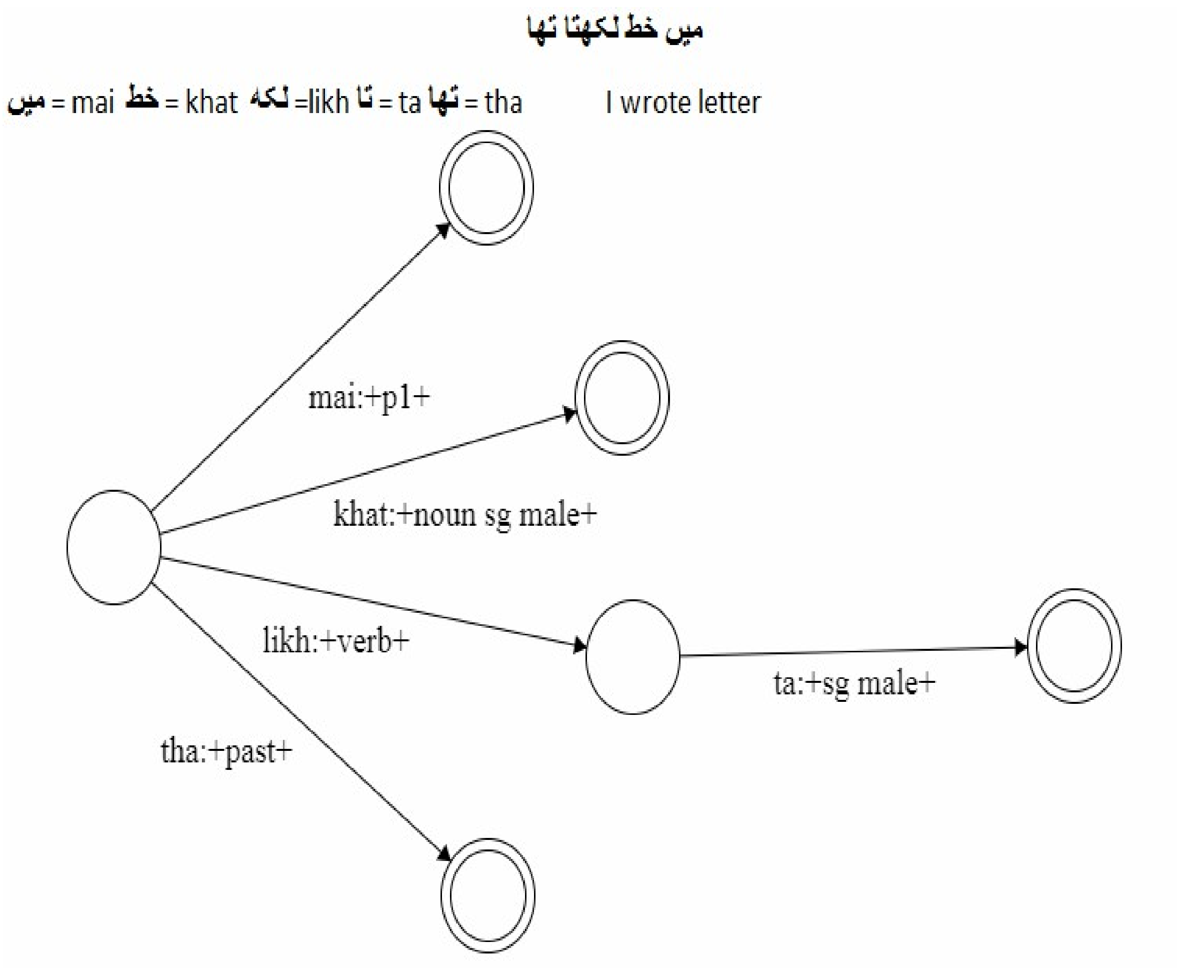
girl works = لڑکی کام کرتی ہے
And
girls works= لڑکیاں کام کرتی ہیں

in both sentences we have کرتی female single and female plural so we can mark it as only +verb+female and to get information of single or plural we can use tense identifier as in above example ہے present single or ہیں present plural(This only exists in simple present)

Coming to the Grammatical person marking in Urdu it is straight forward we mark as follows:

میں:+p1+
آپ،تو:+p2+
وہ،اس:+p3+

So in a nutshell when we have a sentence in Urdu I used to write letters which with Urdu grammer is simple past we have as follows


میں خط لکھتا تھا
میں = mai خط = khat لکه =likh تا = ta تها = tha
I wrote letter
mai:+p1+
khat:+noun sg male+
likh:+verb+
ta:+sg male+
tha:+past+

Conclusion:

This methodology gives a way in which we can make a genaric marking for the Urdu Language but also have some flaws. The flaws are ambiguity present in the Urdu language. For example Urdu is a male dominant language. What I mean by this is suppose we have a female who says wants to say we are eating the sentence would be correct if the group consist of girls but the sentence would change if we have a male in a group.

Female group:	hum khana kha rahi(continuous female) hain
male in a group:	hum khana kha rahay(continuous male) hain

This paper also does not discuess about case system in language which would be done in the future. Using this methodology one can define the order and rules of the language and then can check for correctness of the sentence. Also there is a possibility of implementing a probability estimated spellchecker out of it by using the cost in transition by training and testing the FST.

References

https://www.codecademy.com/learn/learn-python

http://web.eecs.umich.edu/~radev/NLP-fall2015/resources/fsm_archive/fsm.5.html

https://bitbucket.org/coltekin/trmorph-weblicht

https://www.let.rug.nl/vannoord/Fsa/

https://fomafst.github.io/

https://web.stanford.edu/~laurik/publications/ciaa-2000/fst-in-nlp/fst-in-nlp.html

http://primaryessays.blogspot.de/2016/04/tenses-made-easy-urdu-pdf-afzal-anwar.html

http://ling.uni-konstanz.de/pages/home/butt/main/papers/boegeletal.pdf

https://forum.unilang.org/viewtopic.php?t=21245