

Abstract:

Nowadays by improving online shopping resources and e business, many customers prefer to get some online advices, to decide on buying a product or using a service with more confidence. So user feedbacks and comments on different websites are become an important resources for these purposes. and sentiment analysis has make it possible to receive to this goal by working on these resources and give the expected information from them.

sentiment analysis have developed for English language in different aspects, but for Persian language that is the official language of Iran, Afghanistan and some other countries, there isn't many research and methods to make it in use for Persian text and the main reason for that can be the huge difference between formal and informal form of writing in comments and user feedbacks. we have created a Persian sentiment analysis system with the use of GATE and lexicon-based approach that is done for the first time for Persian language. We used processing steps as tokenizing the document, tagging its parts, splitting the sentences and then using jape rules to define each sentence's sentiment. We calculate the sentiment of whole document with using the average sentiment of its sentences. Our system accuracy till now is about 67.7 percentage.

1. Introduction:

Although sentiment analysis is not a new approach to work on documents and texts, but it is not improved a lot for Persian language. On the other hand, number of Persian websites and the people who use these resources in Persian is becoming more and more. This language is used with many people from Iran, Afghanistan, and some other Asian countries and in spite of this fact,

just few researches have done for it in this area and all of the previous works are on machine learning algorithms like svm-based methods.

Persian language has a high level of complexity due to the frequent morphological operations. In addition to complex morphology, Persian language has some other distinctive features, which make it a challenging domain, e.g. morphological complexity, lexicon intricacy, context sensitivity of the script, and free words order due to independent case marking[1]. So different approaches which have done for other languages for example English cannot be used in Persian documents.

For this reason we have created a Persian API service that contains more than 6000 Persian adjectives and phrases with their sentiments and with the use of lexicon-based approach we can define the sentiment of a Persian document with high accuracy in comparison to other different machine learning methods which have done till now. we have tried to gather most common and usual Persian words and phrases in our API to make the result more accurate and useable.

2. Requirements catalogue

2.1. Functional Requirements:

2.1.1. Input Persian text

Description: entered text by user which wanted to have sentiment analysis on it.

2.1.2. Text analysis

2.1.2.1. Tokenize

Description: splits the text into very simple tokens such as numbers, punctuation and words of different types

2.1.2.2. Split sentences

Description: segments the text into sentences that is required for the tagger

2.1.2.3. Tag each part of the text with the use of [2.1]

Description: This produces a part-of-speech tag as an annotation on each word or symbol

2.1.3. sentiment detection

2.1.3.1. Create an algorithm for this step

2.2. Non-functional requirements:

2.2.1. Using lexicon based method to get sentiment of the Persian text.

2.2.1.1. Gathering Persian adjectives and phrases from different resources

2.2.1.2. Implementing a web application for voting Persian people

2.2.1.3. Giving the sense of each adjective depends on the votes they get.

2.2.2. More Non-Functional Requirements

2.2.2.1. Implementation requirements

Description: the application should load from different browsers without any special configuration or settings.

2.2.2.2. Interface requirements

Description: the application should have a good interaction among what user need from the analysis and what the system shows as a result.

2.2.2.3. Operations requirements

Description: application should be responsible if user wants to copy a text from other resources and paste it there to have its sentiment analysis. (Right click inability)

2.2.2.4. Packaging requirements

Description: application should be with its required formats, fonts, libraries... and user shouldn't need to do any additional task to make the text readable or acceptable.

2.2.2.5. Scalability

Description: input text from user would be in the range of 1 – 2000 characters. For huge text user needs to divide it in different parts.

3. Approach:

3.1. Data collection:

In our lexicon-based sentiment analysis system we needed a wide range of Persian vocabularies and their sentiment. As there is not any Persian API, for receiving to this goal, we have gathered about 6000 Persian adjectives and phrases which are used in daily communication between Persian people or use in their written texts and comments in web and on the other hand some of them are from the old usage of language among people but in some documents and text will be useful. Then as we needed the sentiment of each word in this huge resource, we have implemented a web site and asked from different groups of Persian people to visit it and vote on them. When they visited the website they could see the adjectives and phrases with three

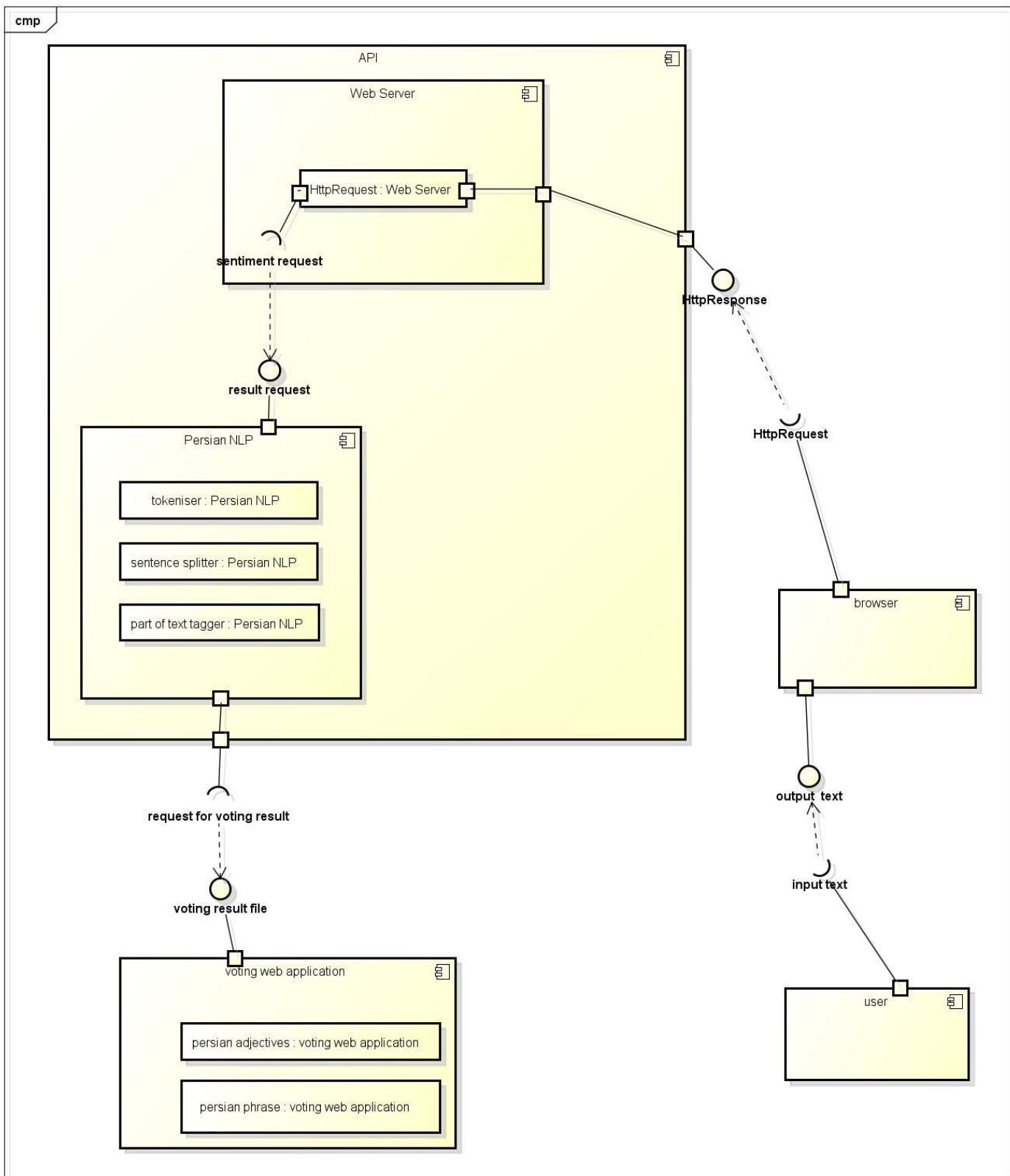
choices for each. If the word is positive, negative or neutral, so depend on their opinion about each word they would define its sentiment.

On the other hand as we wanted our database to be as accurate as possible, we didn't limit each word to be voted just once, we repeated the word in different places in the website and different kind of voters could mention their opinion about that word. So we have each word with different opinions on its sentiment. Then we should get the average for the sentiment of each word, with this method we have defined the adjectives and phrases sentiment in our database for the next steps.

3.2. Process:

In this research we used GATE for natural language processing (NLP). GATE is open source free software and is in active use for all types of computational task involving human language[2].

3.2.1. System structure:



3.2.1. Processing Resources:

Processing resources can be choosing from the list of resources in GATE. For our purpose we have loaded Tokenizer, sentence splitter, pos tagger, Gazeteer, jape rules and Groovy. For the first tree resources (Tokenizer, sentence splitter, pos tagger) we have used Persian plugins that is created by an Iranian researcher [3]. After adding all necessary resources we used simple pipeline to group them together in order and execute them in turn.

3.2.1.1. Tokenizer:

The Tokenizer splits the text into very simple tokens like words, numbers, space and punctuation. As Persian language isn't sensitive for upper and lower case like English, in our tokenizer we didn't mention on it.

3.2.1.3 Sentence splitter:

The sentence splitter, fragments the text into its sentences. The splitter uses a list of abbreviations to help distinguish sentence-marking full stops from other kinds. Each sentence is annotated with the type 'Sentence'.

3.2.1.4 Gazeteer:

In Gazeteer we have added all defined sentiment adjectives and phrases we have gathered and voted before. So our Gazeteer is our sentiment database and wherever it has found those adjectives and adverbs in the text it will put its sentiment tag on it.

3.2.1.5 Jape Rules:

JAPE is a Java Annotation Patterns Engine. JAPE provides finite state transduction over annotations based on regular expressions. [2]

So with jape we can identify regular expressions we have formulated as grammar base. We have two level jape rules, first is in the word level that

means we have created some rules for identifying negative and positive word when they have some special prefix and postfix. For example in Persian, most of the time by coming “با” “Ba” prefix before the noun it becomes positive and with coming

“بی” “Bi” and “نا” “Na” prefixes it becomes negative.

نادرست (Na dorost), با معرفت (Ba marefat), بی ادب (Bi adab), بی اخلاق (Bi akhlagh)

And also we have made some rules for negative verbs too. In Persian we make the verbs negative by using “ن” “n” at the beginning. It has the same meaning as “don't” in English for making the negative verbs.

نمیفهمد (Nemifahmad) > don't understand

ندارد (Nadarad) > don't have

In the second level we have created the jape rules for the whole sentences. in this step number of positive and negative words will be counted and the sentiment of the sentence will be tagged on it with the average sentiment of its words. Then in the second step we focus on the main verb of the sentence, if the verb is tagged as negative with the first level jape rules then the sentiment of the whole sentence will be changed from what it was calculated before. As it is shown in this example:

“این فیلم بازیگران معروف زیادی داشت ولی نتوانست نظر مردم را جلب کند”

“That film had a lot of famous actors but it couldn't attract people's attention.” the negative verb changes the sentiment of the whole sentence as negative. This fact can be true for negative case too. For example:





















“فیلم خوبی نبود”

“It wasn't a good film!”

If the whole sentence's sentiment is negative and the main verb is negative too, the sentiment of the sentence will be recognizing as positive.

“آدم دروغگوئی نیست”

“He is not a liar”

Selected Processing resources		
!	Name	Type
	 Tokenizer	Persian Tokenizer
	 Sentence_splitter	Persian Sentence Splitter
	 POs	Persian POS Tagger
	 Gazetteer	Hash Gazetteer
	 Negation	JAPE Transducer
	 Negword	JAPE Transducer
	 Negverb	JAPE Transducer
	 Sentences	JAPE Transducer
	 Negverb_in_sentences	JAPE Transducer
	 Groovy	Groovy scripting PR

4. Web service implementation:

For implementing the web service, we use Tomcat with Eclipse and we call the pipeline processing resource from GATE.

4.1. Running Tomcat with eclipse [4]:

- **Install Java.** The full JDK (with compiler), not just the JRE (for running existing apps).
- **Unzip Tomcat.** Unzip Tomcat into the location of our choice. Modifying the port so Tomcat runs on 80 instead of 8080, and made two other small changes: enabling directory listings, and making the server automatically restart when faces-config.xml or struts-config.xml changes. These

changes are useful during development, but only the port number change should be set for deployment servers.

- **Install and start Eclipse.** Download from <http://www.eclipse.org/downloads/>. Choose "Eclipse IDE for Java EE Developers", download, and unzip. Double click on eclipse.exe in the eclipse folder, then click on "Workbench".
- **Tell Eclipse about Tomcat.** Click on Servers tab at bottom. R-click, New, Server, Apache, Tomcat v7.0, navigate to Tomcat 7 installation folder (e.g., C:\apache-tomcat-7.0.34), OK.
- **Run Tomcat.** Click on Servers tab at bottom. R-click on Tomcat v7.0, choose "Start". Open <http://localhost/> in a browser. Either way, you will see a 404 error message, but at least the message comes from Tomcat. Then, copy the ROOT app as described in the next section, come back, and reload <http://localhost/> (or <http://localhost:8080/> if using the unmodified version from the Tomcat download site). You should now see a friendly Tomcat welcome page. If you get a "port 80 is already in use" message, go to the Windows Control Panel, Services, and stop the other server (probably IIS). Also, on Linux and Solaris, you need admin privileges to start servers on low-numbered ports like 80. So, if you use the preconfigured Tomcat version but do *not* want port 80, double click on Tomcat at the bottom and change the port from 80 to something else (see "HTTP/1.1" in the "Ports" section on the right). But port 80 is nicer so that you can use URLs of the form <http://localhost/app/blah> instead of <http://localhost:8080/app/blah>, so using 80 should be your first choice.

Ok now Tomcat server work with eclipse.

4.2. Creating a java web service with using Eclipse[5]:

Eclipse by default uses Apache Axis to implement the web service and it provides option to use our choice of web service engine. We decided to go with the default bundled Apache Axis.

4.2.1. Create a Dynamic Web Project

Use the new project from menu and open project wizard. Select 'Dynamic Web Project' and click next. Then give a project name and select a target runtime.

4.2.2. Create a New Runtime

If the run time is not already defined, then click New Runtime and select the version of Tomcat you have installed (already) then click next. Now browse the path of Tomcat home directory and click finish.

4.2.3. Create Web Service Provider Java Class

Create a new package under 'Java Resources – src' named com.SentimentPersian.jee . Then, create a new java class under that package. This is the web service's service provider class.

```
package com.SentimentPersian.jee;  
import java.io.File;  
import java.util.LinkedHashMap;  
import gate.AnnotationSet;  
import gate.Document;  
import gate.Corpus;  
import gate.CorpusController;  
import gate.FeatureMap;  
import gate.Gate;  
import gate.Factory;  
import gate.GateConstants;
```

```

import gate.corpora.RepositioningInfo;
import gate.util.persistence.PersistenceManager;

public class Sentiment {

    public void onCreate () {

        if (Gate.getPluginsHome() != null)
        {
            System.out.println(Gate.getPluginsHome());
        }
        else {
            //set the address of gate plugins
            Gate.setPluginsHome(new File("C:/Program
Files/GATE_Developer_8.0/plugins"));
        }
    }

    public String persian_sentiment(String text) throws Exception {

        System.gc();
        //if(Gate.getGateHome() != null) { throw new IllegalStateException(
            //      "gateHome has already been set it"); }
        //Gate.setGateHome(new File("C:/Program Files/GATE_Developer_8.0"));

        onCreate();

        //Set the address of gapp file of project here
        File PersianGapp = new File( "C:/Users/mohammad/Desktop/New
folder/Gate/application.xgapp");
        // initialise GATE - this must be done before calling any GATE APIs
        Gate.init();
    }
}

```

```
// load the saved application
CorpusController application =
    (CorpusController)PersistenceManager.loadObjectFromFile(PersianGapp);

// Create a Corpus to use. We recycle the same Corpus object for each
// iteration. The string parameter to newCorpus() is simply the
// GATE-internal name to use for the corpus. It has no particular
// significance.
Corpus corpus = Factory.newCorpus("BatchProcessApp Corpus");
application.setCorpus(corpus);

// process the files one by one

// load the document (using the specified encoding if one was given)

Document doc = Factory.newDocument(text);

// put the document in the corpus
corpus.add(doc);

// run the application
application.execute();

String featureName = "Doc_sentiment";
FeatureMap features = doc.getFeatures();
// remove the document from the corpus again
corpus.clear();

// doc.getFeatures().
// Release the document, as it is no longer needed
Factory.deleteResource(doc);
```

```

        LinkedHashMap originalContent = (LinkedHashMap )
            features.get(featureName);

        String obj =(String) originalContent.get("sentiment");
        //obj =obj+(String) originalContent.get("positive");
        //obj =obj+(String) originalContent.get("negative");
        //System.out.println(obj);

        //System.out.println("All done");
        return obj;
    }
}

```

With this step everything is complete. If you have Eclipse it will take care of all the remaining work like creating web service, generating wsdl, skeleton, service client, stub and etc.

4.2.4. Create a Web Service

Now the service class is ready and we need to create a web service using this java class.

Right click on 'Java Resources' -> New and select 'Web Service' under 'Web Services' folder from the wizard. Click Next button. Select Service

Implementation:

In this Web Service wizard, use the browse button and select the java class written earlier, which is our service implementation java class. Then, drag the slider bar to upper most in service and client part, then; enable the Publish the Web service check box.

✓ Web Service Client Creation

Added to configuring the service implementation, we are instructing Eclipse to generate a web service client also. This will create a dynamic web java project and create a web service client for the web service created. Instead we can also create a java project and write a client to access the web service.

- ✓ Deploy Web Service and Client

Click Next till the Server Startup wizard and then click Start Server. This step will start the associated runtime Tomcat.

5. Testing:

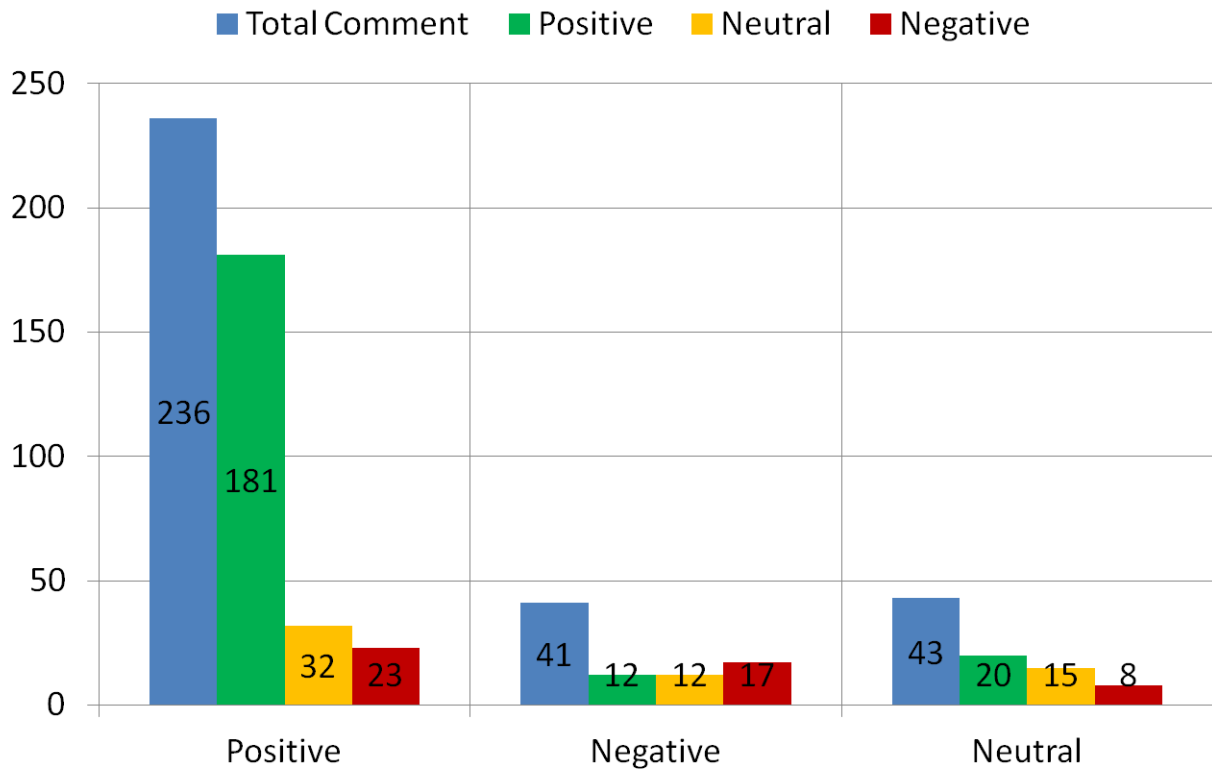
5.1. Black box testing:

5.1.1. Unit testing:

Focus on performance of each component if works properly. For example here we check the accuracy of tokenizer, sentence splitter and pos tagger

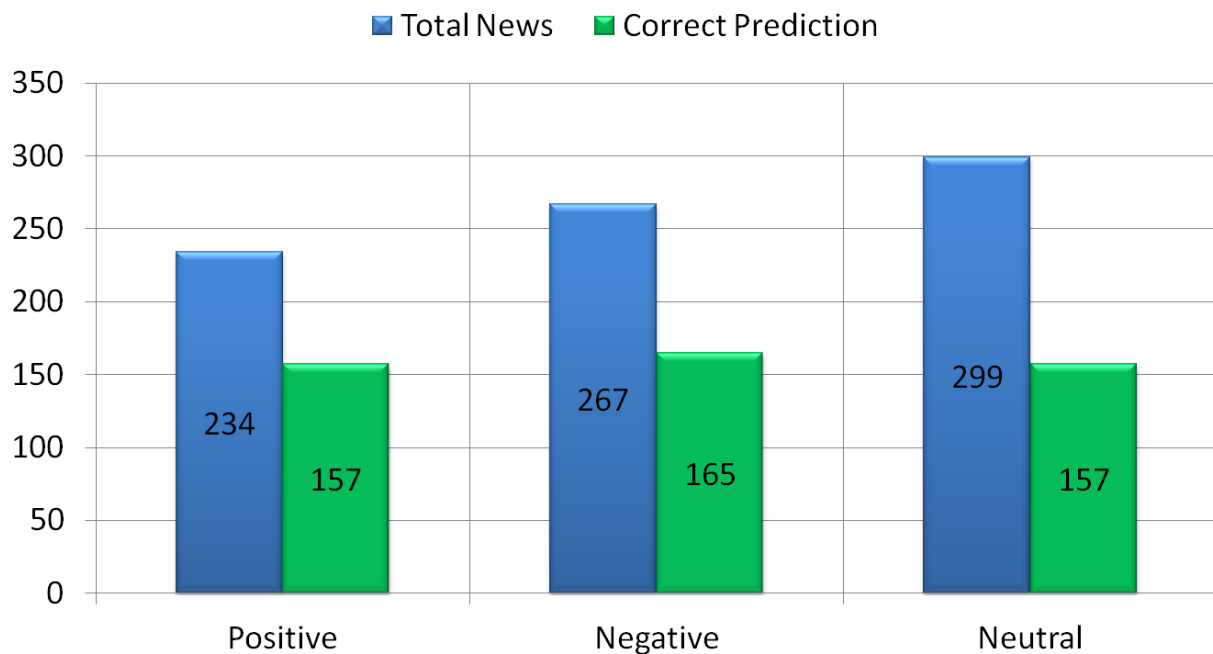
5.1.2. Performance testing:

To test the performance of our system, we choose a Persian website' comments (www.iran-booking.com) . This website is about hotel reservation and a lot of Persian people wrote about their opinions and experience about using different hotels in different places. The comments in this website are star base and each comment depends on its information has different number of starts from 1 to 5. We got about 310 comments from this star base comments and identifying 1,2 stars as negative,3 starts as neutral and 4,5 stars as positive text, and these comments made our performance database test data. Then we give these comments to our system and calculate the sentiment of each. by comparing the sentiment we reach from each comment to its star base sentiment we give the accuracy of 66.6 percentage from our system. The chart below show the result of this test:



Figur1. Performance test result1

For the next level of performance test we choose another method of testing. in this method we searched between different famous Persian news websites and chose the most four popular of them as “farsnews.com”, “tabnak.ir”, “yjc.ir” and “varzesh3.com”. We got about 5000 news from different categories of sport, social, politics, economic and international and chose 800 of them randomly. Then we upload these 800 news to our website and ask Persian people to put their opinions about each news as positive, negative and neutral. Here, we gathered about 1200 voted on those 800 news. For increase of accuracy some of them have repeated too. Then with comparing these voted result by the result our system we receive 62 percentage of accuracy with this method. The chart below shows the details:



Figur2. Performance test result2

5.2. Server response time:

In the system with windows 8.1, 3 Gigabyte Ram, CPU cor i5 2.5 GHZ, 500 request with average of 3000 character take about 900 seconds for response.

5.3. Integration test:

For integration testing we use Big bang integration method. In Big Bang integration all components or modules are integrated simultaneously, after which everything is tested as a whole. So first we have testes tokenizer, sentence splitter and pos tagger in a document and then we test all together. The image below, shows the tokens of a text:

GATE Developer 8.0 build 4825

File Options Tools Help

GATE

Applications

Persian Sentiment Analysis

Language Resources

t3

test

Processing Resources

Groovy

Negverb_in_sentences

Sentences

Negverb

Negword

Negation

Gazetteer

Messages

Persian Sentiment...

t3

Annotation Sets

Annotations List

Annotations Stack

Co-reference Editor

Text

شماری از گروه های وابسته به جبهه النصره و داعش وارد شهرک مرزی عرسال در داخل خاک لبنان شدند و ارتش لبنان با این گروهک ها وارد درگیری شد. به گزارش تسنیم به نقل از پایگاه خبری جدید؛ شماری از نیروهای تروریستی وابسته به دولت اسلامی در عراق و شام داعش به اتفاق شماری از اعضای تروریستی جبهه النصره لحظاتی قبل وارد روستای جرد العرسال در مرز سوریه و لبنان در داخل خاک لبنان شدند. منابع خبری از داخل این روستا اعلام کردند تروریست های مسلح به سمت اهالی در حال تیراندازی هستند و مانع از خروج اهالی از این روستا به مناطق دیگر می شوند و عملاً اهالی روستا را به عنوان سیر انسانی قرار داده اند. برخی از اهالی که موفق به خروج از روستای عرسال شده اند در این باره اعلام کردند: گروه های تروریستی داعش و جبهه النصره به سمت ما تیراندازی کردند و اجازه نمی دادند از این روستا به دیگر مناطق لبنان متغیر شویم. هم اکنون نیروهای نظامی ارتش لبنان با گروه های تروریستی وابسته به داعش و جبهه النصره در خیابان های این روستا درگیر هستند. منابع خبری از کشته شدن 10 شهروند لبنانی در شهرک عرسال خبر دادند همچنین گفته می شود که 13 شهروند لبنانی توسط عناصر مسلح به گروگان گرفته شده اند. دهها تروریست وابسته به جبهه النصره و داعش سعی دارند به یکی از پایگاه های نیروهای وابسته به ارتش لبنان در شهرک عرسال یورش آورند. اما نیروهای نظامی ارتش به شدن در حال مقاومت از این پایگاه هستند. از سوی دیگر اهالی عرسال در لبنان اعلام می کنند گروه های مسلح تروریست به سمت اهالی در این شهرک تیراندازی می کنند. همچنین گزارش شده 13 نظامی دیگر ارتش لبنان به دست تروریست ها روده شدند.

Lookup

Sentence

SpaceToken

Split

☒

Token

negverb

Original markups

Type	Set	Start	End	Id	Features
Token		1	6	2	{category=QUA, kind=word, length=5, string=شماری, type=farsi}
Token		7	9	4	{category=P, kind=word, length=2, string=ار, type=farsi}
Token		10	18	587	{category=N_PL, kind=word, length=8, rule=Suffixes, string=های گروه, type=farsi}
Token		19	25	10	{category=ADJ_SIM, kind=word, length=6, string=وابسته, type=farsi}
Token		26	28	12	{category=P, kind=word, length=2, string=به, type=farsi}
Token		29	33	14	{category=N_SING, kind=word, length=4, string=جبهه, type=farsi}
Token		34	40	16	{category=N_SING, kind=word, length=6, string=النصره, type=farsi}

280 Annotations (0 selected) Select

Document Editor

Initialisation Parameters

Relation Viewer

GATE Developer 8.0 build 4825
File Options Tools Help

GATE
Messages t3 Persian Sentiment...

Annotation Sets
Annotations List
Annotations Stack
Co-reference Editor
Text

سلام هتل خیلی خوبی بود مخصوصاً مدیریت رستوران آقای سبحان بسیار مرد خوبی بود روشن های هتل خیلی خوب نبودن ولی کلاً هتل خوبی بود صبحانه و ناهار خیلی خوب بود یا اینکه بچه همراه ما بود غذاهایش خوبی بود

☒ Lookup

☒ Sentence

☒ SpaceToken

☒ Split

☒ Token

Original markups

Type	Set	Start	End	Id	Features
Sentence		0	195	82	{neg=0, pos=9, score=18, sentiment=positive}
Token		0	4	1	{category=N_SING, kind=word, length=4, string=سلام, type=farsi}
SpaceToken		4	5	2	{kind=space, length=1, string=}
Token		5	8	3	{category=N_SING, kind=word, length=3, string=هتل, type=farsi}
SpaceToken		8	9	4	{kind=space, length=1, string=}
Lookup		9	13	83	{majorType=intens}

88 Annotations (0 selected) Select

New

Doc_sentiment {sentiment}

MimeType text/plain

docNewLineType

gate.SourceURL file:/C:/

Resource Features

Document Editor

Initialisation Parameters

Relation Viewer

Persian Sentiment Analysis run in 0.45 seconds

ENG 1:28 AM
10/1/2014