TRANSLATION WITH USER COMPUTATION

by

Salih Can Aydoğdu

B.S, Computer Engineering, Yeditepe University, 2006

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2009

TRANSLATION WITH USER COMPUTATION

APPROVED BY:

Dr. Suzan Üsküdarlı (Thesis Supervisor)	.....
Assist. Prof. Haluk Bingöl	.....
Assoc. Prof. Yağmur Denizhan	.....

DATE OF APPROVAL: 17.09.2009

ACKNOWLEDGEMENTS

With my deepest gratitude; I would like to thank to my thesis supervisor Dr. Suzan Üsküdarlı for her guidance, great patience and endless support all through this thesis. I am grateful to Assist. Prof. Haluk Bingöl and especially Assoc. Prof. Yağmur Denizhan for participating in my thesis jury, their criticism and helpful comments that have greatly improved this thesis.

I am specially indebted to my family, Mustafa Aydogdu, Zuhal Aydogdu and my brother Onur Aydogdu for their unconditional love and endless support throughout my life. I am also grateful to Büsra Sağocak for her endurance and tenderness.

Also I would like to express my gratitude to BAP Commition for supporting this study with projects 09HA102P and 08A103.

Lastly I would like to express my gratitude to TUBITAK for supporting me with National Scholarship Programme for M.Sc. Students - 2210.

ABSTRACT

TRANSLATION WITH USER COMPUTATION

Human Computation aims to solve problems by means of combining the power of human and computer computation. The typical approach is to assign tasks to computers and humans based on what they are better capable of performing. The aim is to bring human-computer relationship into a symbiotic mutualism, where humans compute what they are good at and vice versa.

Games with a purpose introduced a concept that harvests human brain power. They try to solve a large scale complex problem with a game structure. In our work, we present a model with an objective of improving machine translation. TURC, Translation with User Computation, is a web-based multiplayer online game.

TURC focuses English-Turkish translation. As there are large amounts of English data available in the web, the lack of Turkish is disappointing. There are available machine translation efforts between these languages, such as Google Translate. However a speaker of both languages can easily tell that these efforts are insufficient. Our model is designed to generate output to help improving machine translation while making people enjoy translation with the element of competition. Two non-communicating players are presented a machine translation of an English sentence. They have the options of adding, editing, deleting or moving a word available. Their aim is trying to make the same actions. A score value is generated according to the correspondence of these actions. A prototype of our game model is hosted on a server for two weeks and the results are discussed.

ÖZET

KULLANICI HESAPLAMASIYLA ÇEVİRİ

Bazı problemler bilgisayarlar için çok zor, insanlar içinse kolaydır. İnsan bilgisayar ilişkisini karşılıklı fayda temeline, insanların kendi iyi yaptıkları işleri, bilgisayarlarınsa kendi iyi yaptıkları işleri yapmalarını sağlayarak dönüştürebiliriz. Bilgisayar oyunları, insan hesaplamasında kullanılan, önde gelen bir motivasyon aracıdır.

Bir amaç için oyunlar, insan hesaplamasını kullanma adına ortaya atılmış bir konsepttir. Bunlar eğlenceli küçük oyunlardır. Oyunlar oynanırken arkaplanda büyük kompleks bir problemin çözümüne katkıda bulunurlar. TURC, Kullanıcı Hesaplamasıyla Çeviri, web üzerinden oynanan çok kişilik bir oyundur.

TURC İngilizce'den Türkçe'ye çeviriye odaklanmaktadır. İnternet üzerinde çok sayıda İngilizce kaynak olmasının yanında, Türkçe kaynaklar oldukça sınırlıdır. İki dil arasında Google Translate gibi makina çevirisi programı denemeleri mevcuttur. Fakat iki dili de iyi konuşan biri için, bu programların çevirilerinin yetersizliği aşikardır. Bizim modelimizde, rekabet faktörüyle oyuncular eğlenirken, kolay anlaşılabilen bir arayüz aracılığıyla çıktı olarak makina çevirisini geliştircek önerilerde bulunmaktadırlar. Birbiriyle iletişimi olmayan iki oyuncuya İngilizce bir cümlenin makina çevirisi sunulur. Kelime ekleme, çıkarma, düzenleme ya da yer değiştirme seçenekleri mevcuttur. Oyuncuların hedefleri birbirleriyle aynı değişiklikleri yapmaktır. Bunun en kolay yolu da çeviriyi düzeltmekte yatar. Oyuncuların hamlelerinin benzerliği baz alınarak bir skor üretilir. Bu sistemin uygulandığı bir prototip internette sergilenmiştir. Bu çalışmada oyun modelimiz ve çalışmalarımızın sonuçları anlatılmaktadır.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . iii
ABSTRACT . . . iv
ÖZET . . . v
LIST OF FIGURES . . . ix
LIST OF TABLES . . . xi
LIST OF SYMBOLS/ABBREVIATIONS . . . xii
1. INTRODUCTION . . . 1
1.1. Motivation . . . 1
1.2. Proposed Solution . . . 3
1.3. Outline . . . 4
2. BACKGROUND . . . 5
2.1. Human Computation . . . 5
2.1.1. Incentives . . . 5
2.1.2. Methods . . . 6
2.2. Games with a Purpose . . . 7
2.2.1. ESP Game . . . 8
2.2.2. Peekaboom . . . 10
2.2.3. Verbosity . . . 10
2.2.4. Tag a Tune . . . 11
2.2.5. Phrase Detectives . . . 12
2.2.6. Foldit . . . 12
2.3. Language Translation Tools . . . 12
2.3.1. Google Translate . . . 12
2.3.2. Caitra . . . 13
2.3.3. WikiBABEL . . . 14
2.3.4. Online dictionaries . . . 14
3. TURC: TRANSLATION GAME . . . 15
3.1. Requirements . . . 15
3.2. The Game Model . . . 16

3.2.1. Game Flow . . . 17
3.2.2. Player Actions . . . 19
3.2.3. Scoring Mechanism . . . 21
3.2.4. Trust . . . 22
3.2.4.1. Cheating . . . 22
3.2.4.2. Player Levels . . . 23
3.3. Discussion . . . 23
3.3.1. How to make a game enjoyable? . . . 23
3.3.2. How to make a game productive? . . . 24
4. PROTOTYPE . . . 26
4.1. Software Environment . . . 26
4.2. System Source . . . 27
4.2.1. Text to be translated . . . 27
4.2.2. Translator . . . 28
4.3. User Interface . . . 29
4.3.1. How to play? . . . 29
4.3.2. PLAY . . . 30
4.3.3. HI-SCORES . . . 31
4.3.4. About . . . 31
4.4. Game Interface . . . 31
4.4.1. North Part . . . 31
4.4.2. Center Part . . . 32
4.4.3. West Part . . . 32
4.4.4. East Part . . . 32
4.4.5. South Part . . . 33
4.4.6. Dictionary helper . . . 34
4.5. Server Client Integration . . . 34
4.5.1. Multiplayer Game Infrastructure . . . 36
4.5.2. Player Synchronization . . . 37
4.5.3. Single-player System . . . 37
4.5.4. Two-player System . . . 38
4.6. Score System . . . 38

4.6.1. Adding a new word 38
4.6.2. Deleting a word 39
4.6.3. Editing a word 39
4.6.4. Order of the words 39
4.6.5. Prerecorded Game Points 40
4.7. System Output 40
5. Experiment and Results 42
5.1. Setup 42
5.2. Evaluation Metrics 42
5.2.1. Throughput 42
5.2.2. Enjoyability 43
5.2.3. Expected Contribution 43
5.3. Results 44
5.3.1. Statistics 44
5.3.2. Evaluation Metrics 45
5.3.3. Example Results 45
5.4. Discussion and Future Work 49
6. CONCLUSIONS 51
REFERENCES 53

LIST OF FIGURES

Figure 1.1. The percentage of publicly accessible websites by language 1

Figure 1.2. Top 10 Internet Languages listed in www.internetworldstats.com . 2

Figure 2.1. An example output-agreement game 8

Figure 2.2. The partners are agreeing on a label 9

Figure 2.3. A sample inversion-problem game 10

Figure 2.4. A sample input agreement game 11

Figure 2.5. Unfolded and folded protein in-game screenshots 12

Figure 2.6. User interface of Caitra 14

Figure 3.1. TURC Game Flow Algorithm 18

Figure 3.2. TURC Player Actions 20

Figure 3.3. TURC Score Algorithms 22

Figure 4.1. Prototype Overview 26

Figure 4.2. A screenshot from How to play? screen 29

Figure 4.3. A screenshot from PLAY tab before login 30

Figure 4.4. A screenshot from Hi-Score screen 31

Figure 4.5. A screenshot of the game interface 32

Figure 4.6. a. Edit b.Wrong Translation c. Untranslated d. Doesn't Needed e.Add 33

Figure 4.7. Dictionary helper and autocomplete textbox in action 34

Figure 4.8. A diagram representing the finite state machine of the prototype . 35

LIST OF TABLES

Table 2.1. Division of labor in computation (Selection agent vs. Innovation agent) 6

Table 5.1. Number of player agreements on improvements over two weeks . . 44

LIST OF SYMBOLS/ABBREVIATIONS

AJAX	Asynchronous JavaScript and XML
ALM	Average Lifetime Play
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart
GWAP	Games With A Purpose
GWT	Google Web Toolkit
NIO	New Input Output
TURC	Translation with UseR Computation
XML	Extensible Markup Language

1. INTRODUCTION

1.1. Motivation

Internet is everywhere. It is in our houses, in our schools, in the cafe we drop by for a coffee break, even in our cellphones, with immediate access practically everywhere we go. This revolutionized many lives, giving a chance to access lots of information about anything from anywhere within seconds. However, this is not the case for everyone who access Web. The language of Web content is widely varied with a great majority being in English (see Figure 1.1 [1]).

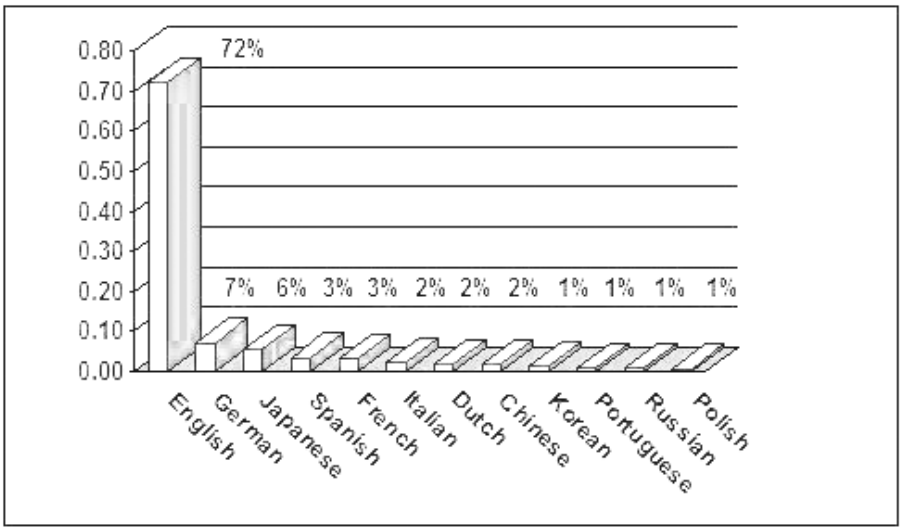


Figure 1.1. The percentage of publicly accessible websites by language

For example in July 2009, Wikipedia [2], an online encyclopedia created by volunteers, has over three million English articles. The number of Turkish articles are only 135,209. This means that available Turkish content is roughly 5% of the English content in Wikipedia, furthermore the Turkish articles contain only part of the corresponding English articles. So a Turkish only speaking person has access to only a small portion of information available through Wikipedia due to the language barrier. A similar situation is the case throughout the Web.

Organizing world's information and making it universally accessible and useful is a big challange. Translation is essential to accomplish this mission, because much of the information is not universally accessible. For web content that is available in some language, translation to other languages would be most beneficial. The sheer volume of content available makes it necessary to cope with translation in an effective and sustainable manner. Thus, machine translation efforts [3–5] have been underway with varying degrees of success. However, there is still a large room for improvement. Most of the machine translators require bilingual text corpuses, which is only available by translations of professional human interpreters. This process is too costly and depending on the number of interpreters working on such corpuses it is usually too slow.

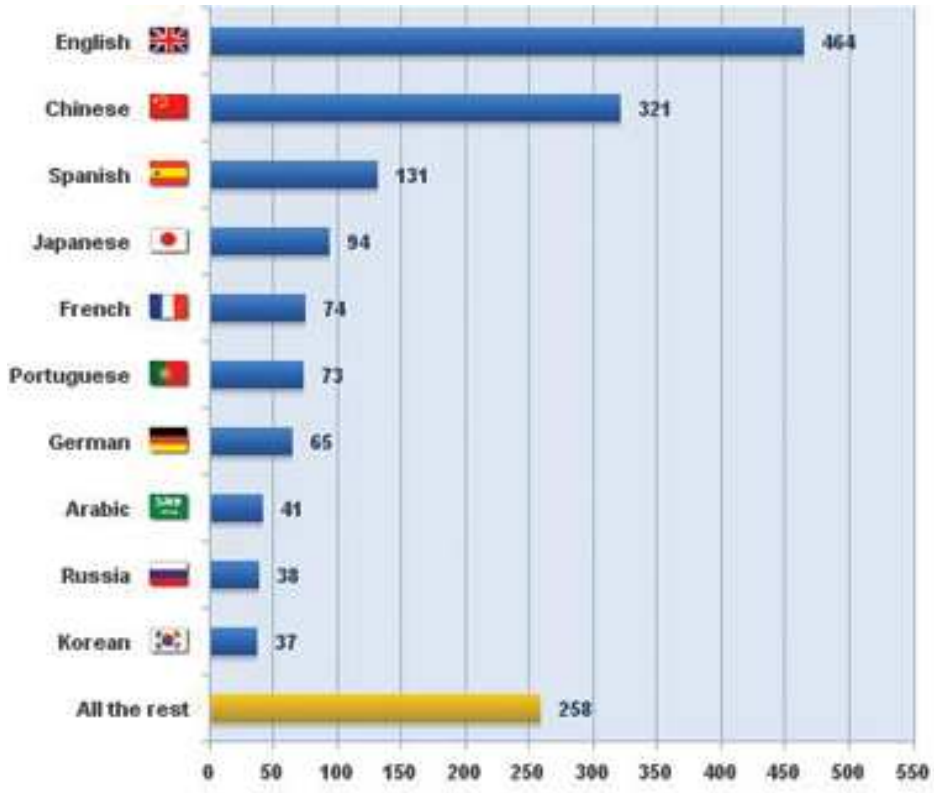


Figure 1.2. Top 10 Internet Languages listed in www.internetworldstats.com

A supplemental approach to translation could be to harvest the voluntary efforts of people willing to translate. For this a human computation model in the form of a game could be used for translation. The Entertainment Software Association reported

that more than 200 million hours a day spent on computer and video games only in U.S. [6]. Most of the time such games serve little beyond the enjoyment of the player. This presents an opportunity for harvesting a part of this energy towards useful outcomes. A game format could be used to motivate users for translation. The outcome of such a game could be used to improve machine translators as well as translations themselves.

1.2. Proposed Solution

Harvesting the energy people spent for playing games is our starting point. This is an opportunity that is even much more relevant with respect to tasks that people are really good at and computers are not. With this in mind the notion of games with a purpose [7] emerged. Such games fall under human computation, which aims to solve problems with human intervention. The general idea is to let computers do what they are good at and people do what they are good at.

Language translation is one of the topics that people are better than computers. Identifying the mistakes in a given translation is much easier than translation for a native speaker. This is possible even if the subject only sees the translated text. Grammar mistakes and unsuited words can be well noticed.

To explore human translation we developed a game that simultaneously presents some text and its machine translation to two players who try to correct the machine translation. This is a cooperative game where players are awarded points based on the similarity of their corrections.

In this thesis we present a game with a purpose model and prototype for improving machine translations from English to Turkish languages. The results gathered from the deployed prototype are also presented.

1.3. Outline

In Chapter 2, you will find explanation about previous research and complementary information for other chapters. In Chapter 3 you will find a description of the game model. Chapter 4 elaborates on the very finest details of the game implementation and the overall system that encapsulates the game. In Chapter 5, we describe the experiment's setup and the results obtained after the experiment, and the evaluation. In Chapter 6, we draw conclusions and talk about some further research topics to be pursued.

2. BACKGROUND

This work is based on human computation in the form of a game with a purpose. This chapter describes the human computation and games with a purpose. The application area of translation is also discussed.

2.1. Human Computation

Human computation is a computer science technique in which programs harness human abilities to solve large scale, complex problems. In traditional computation, a computer is employed to solve a problem by a human who provides a formalized problem description and recieves a solution to interpret. In human computation, the roles are frequently reversed or shared. A computer asks humans to solve a problem, collecting, interpreting and integrating their solutions [8].

Human computation combines intelligent abilities of many people to solve large scale problems by allowing them to perform simple operations on available knowledge such as select, contribute, modify and recombine pieces of knowledge.

2.1.1. Incentives

There is a motivation point for humans to become a part of a human computation process, such as:

- Cheating: Some people join the process just to test the system stability and try beat associated rules.
- Curiosity: Some people join the process to see what this thing is all about.
- Entertainment: People are enrolled in the process, simply because they are having fun. Especially computer games use this approach.
- Esthetic satisfaction: Desire for esthetic satisfaction.
- Knowledge: Sharing knowledge and communication is a motivation point in most

Table 2.1. Division of labor in computation (Selection agent vs. Innovation agent)

	Computer	**Human**
Computer	Genetic Algorithm	Interactive genetic algorithm
Human	Computerized Tests	Human-based genetic algorithm

of processes.

- Income: Some projects pay money or give prize to its users to harness their computational power.
- Reciprocity: Helping each other and being a part of a community is an incentive for some people.
- Research: Results are beneficial to some people for research purposes.
- Socialize: Meeting new people, having a different activity in daily life are some of the incentives for some people.
- Volunteerism: Some people enroll just because they support the cause of the project

2.1.2. Methods

Human computation combines intelligent abilities of many people to solve large scale problems by allowing them to perform simple operations such as select, contribute, modify, and recombine. We can think that modification corresponds to mutation, thus these systems can be explained by evolutionary algorithms.

Computation can be classified in four different topics, regarding the division of labor. Selection agent, performs actions to select and innovation agent performs the computation and applies changes. (See Table 2.1 [8])

Computation involving human contribution:

- Computer-Human: The computation is done by the computer, the human involment supplies the input and the output is also generated for humans. Interactive Evolutionary Computing [9] introduces a system where humans select images among several and according to those selection an abstract drawing is generated. Computer is making the innovation by combining images, and human becomes the selection mechanism (fitness condition). A witness, trying to identify a suspect can select face pictures, which will be later merged to construct a virtual face of the suspect.
- Human-Human: The computation is done by humans, the input is supplied by humans and the output is also generated for humans. Wikipedia [2] is a project of creating a free online encylopedia by user contribution. Contributing new page and its incremental edits are the available actions. Here humans are making both the innovation and the selection. Also social search applications, such as digg [10], fall in to this group. They gather information from its users and recommend the most referred (fittest) contibution.
- Human-Computer: The computation is done by humans, the input is supplied by the computer and the output is also for the computer. Computerized tests such as CAPTCHA [11] and reCAPTCHA [12] are used to seperate humans from computers by presenting a problem that is easy for humans but difficult for computers. reCAPTCHA introduces an additional system to help digitize books by presenting words from scanned old books that optical character recognition tools cannot decipher. Also games with a purpose(GWAPs), are programs that extract knowledge from users in an entertaining way. They will be detailed in the next section.

2.2. Games with a Purpose

People tend to play games in their free time. There are several very successful interactive online games on the web. A leading example is a learning bot which is used in Q20 [13], an online game which has reached over 70 million game sessions. Another example is the massive multiplayer online game of Monopoly that is played using Google Maps had 1.7 million distinct users in its launch day [14]. These numbers

show us the huge potential of human computational power that can be harvested from computer games.

Games with a purpose [7] is a concept introduced by Luis von Ahn from Carnegie Mellon University addressing this issue. These are games that people enjoy playing and at the same time unintentionally being a part of larger computation. There are several examples, which are explained in the following sections.

2.2.1. ESP Game

ESP Game [15] is a revolutionary project that has introduced a new game genre based on human computation which is called games with a purpose. This game aims to label images in order to categorize the massive amount of pictures that are on the Web.

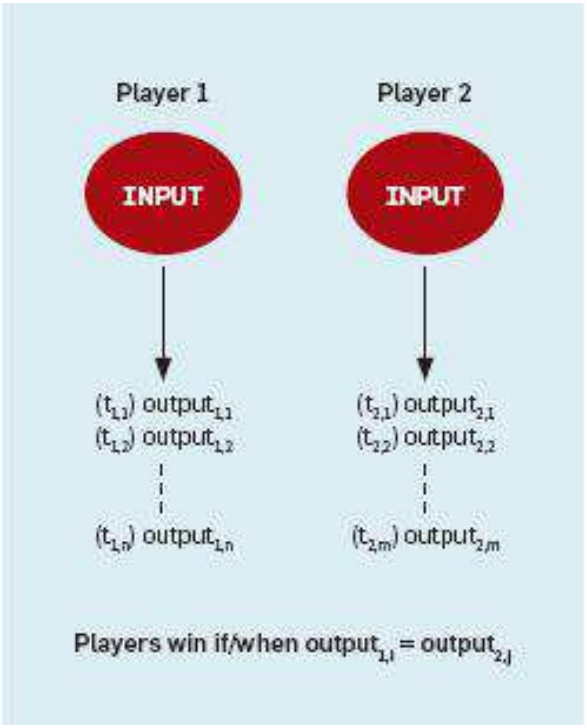


Figure 2.1. An example output-agreement game

It is a simple web browser game. Two players who have no communication in between are presented the same series of images for two minutes. This game is based on

output agreement. Each player has the ability of entering words or asking to skip the image. Their aim is to enter the exact same word for as much as different images. They have only the shown image in common, so they enter words related with the image as the easiest way of agreement. If any of the words entered by the users match each other, both users are awarded some points. This structure is called output-agreement (See Figure 2.1 [16]). If both agree to skip, only the next image is presented. Less common words that match gives more points, because it is hard to match rare words with just an image in between. And a rare word better describes the image.

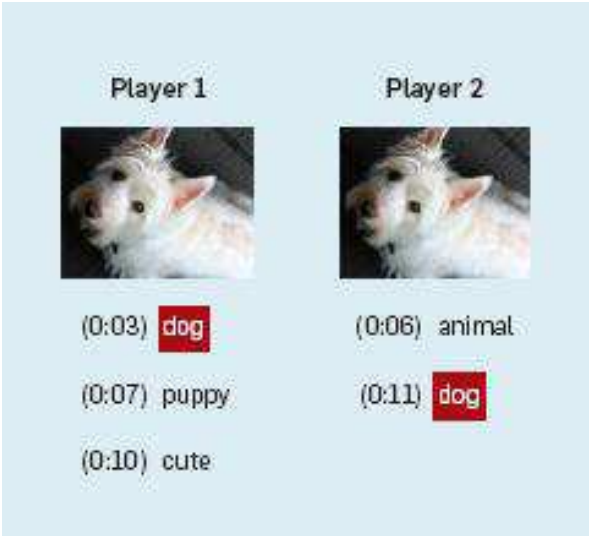


Figure 2.2. The partners are agreeing on a label

Matching words are considered as candidates for identifying these pictures. The output of the game is these words. This output can be later processed according to the majority of the words matched related to the image, later to be used as the identifiers of those images called tags. This process is later added to the game as an input. If an image is already related with a word (tagged), that word is considered as a taboo word. A word that is not allowed for a player to type. Players should match on other words for that image which helps to increase the total number of tags the image has [15].

2.2.2. Peekaboom

This game is similar to ESP game. However, instead of depending on the output agreement, this game introduces an inversion problem approach(See Figure 2.3 [16]).

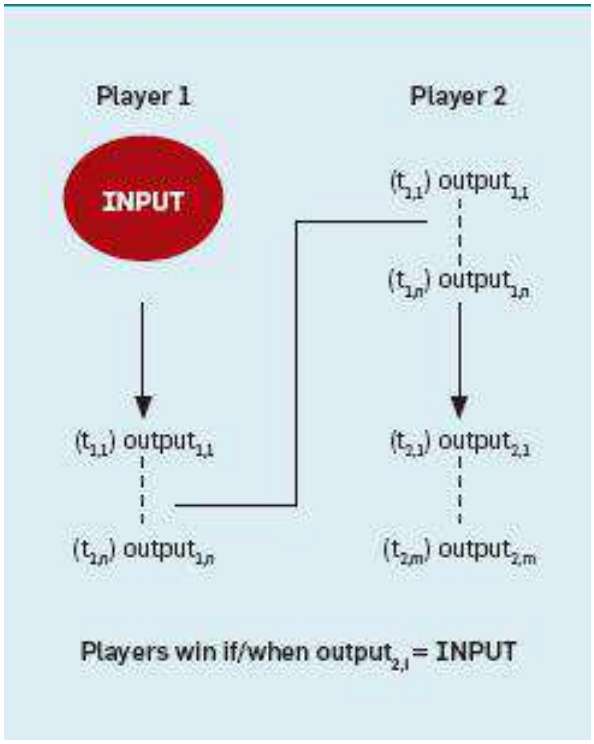


Figure 2.3. A sample inversion-problem game

In this case, one player identifies a part of a given picture by marking a region, which is revealed to the other player, who now has to guess, what it is. Using this approach, regions within the pictures are identified [17].

2.2.3. Verbosity

In the Verbosity game [18], is much like the popular taboo game. The aim is to gather common sense knowledge. One player (the narrator) receives an input word and using pre-supplied template sentences must accurately describe that word sufficiently well so that the second player(the guesser) can guess it correctly. Each player takes turns in narrating and guessing. Multiple players agreeing on terminology and contexts

ensure accuracy. The game is quick at only four minutes in which players go through as many words as possible [18].

2.2.4. Tag a Tune

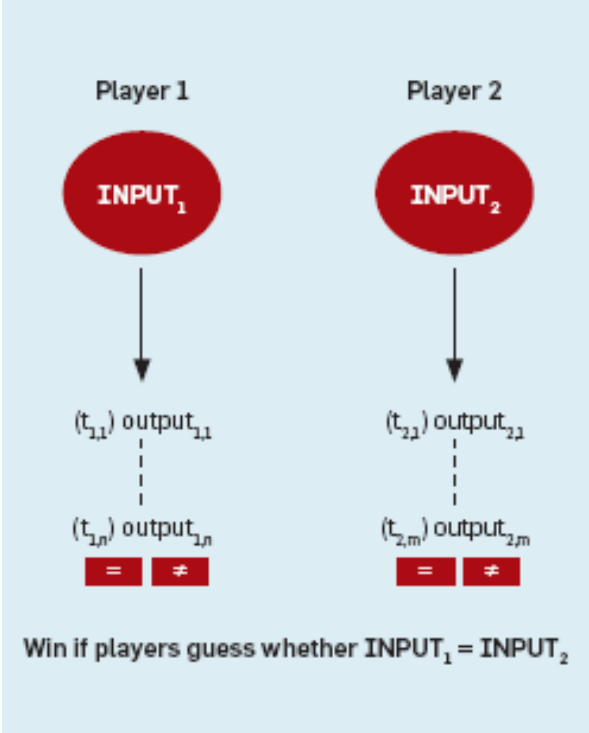


Figure 2.4. A sample input agreement game

Tag a Tune [19] introduces the input agreement game structure(See Figure 2.4 [16]). In this game, each player listens a song and using words, tries to describe the sort of music they hear. Fast, slow, baroque, violins, techno, opera, rock, etc. Based on the descriptions they see from their unseen partner, they each must decide whether they are listening to the same or a different piece of music. Meanwhile, the machine is learning that this music file has these attributes, cross-checked by multiple human game players.

2.2.5. Phrase Detectives

Phrase Detectives is an approach developed by Essex University. This game assigns anaphoric links within the text. This games ask questions, and create a new question from the answer to ask other users and verify. This decision validation process is used to gather judgements from users as well as to provide their score results. Their output can be used in text summarization [20].

2.2.6. Foldit

Foldit [21] is a unique approach to games with a purpose. It is indeed a desktop application. However this makes using 3D technology possible which is essential for the project.

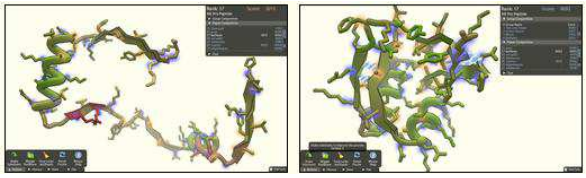


Figure 2.5. Unfolded and folded protein in-game screenshots

The aim of this project is to figure out which of the many possible protein structure is the best, and does it by taking advantage of humans' puzzle solving intuitions and having people competitively to fold the best proteins. This process is considered to be one of the hardest problems of biology and even for computers it would take a lot of time and money doing this [22].

2.3. Language Translation Tools

2.3.1. Google Translate

One of the Google's beta products, Google Translate [3] makes translations between 51 languages. Unlike usual rule based approach, this system takes a statistical

approach. They feed the computer billions of words of text, both monolingual text in the target language, and aligned text consisting of examples of human translations between the languages. Then they apply statistical learning techniques to build a translation model [23].

Google Translate has also an embedded feedback mechanism, by which users suggest improved translations, that are later used to improve the translation.

Google has developed Google Translator Toolkit [24] to use their translation facilities in a more user friendly environment. Using this service you can translate documents stored on your computer, web pages, Wikipedia articles and Knol articles. After importing a document, Google generates the translation, displaying it next to the original text. You can select a sentence from the original document and Google lets you edit the translation.

Google Translator Toolkit has social features too. You can invite other people to help you translate collaboratively. However the system doesn't display the collaborators that are currently editing a document. Also a new feature called "translation memory" stores all the translations made using this system. If a similar sentence as you are translating is found in the database you can use it as a template, you can also rate the translation.

2.3.2. Caitra

Caitra [25] is a novel tool that aids human translators. Its name is generated from computer aided translation. It has the abilites of making suggestions for sentence completion in an interactive machine translation setting, providing alternative word and phrase translations and allowing users to post-edit machine translation output.

In this system user uploads the file he wants to translate and Caitra assists him through the translation. Unfortunately the system recognises only translations between English to French, German, Spanish and Thai with French, German, Spanish



Figure 2.6. User interface of Caitra

to English.

2.3.3. WikiBABEL

WikiBABEL [26] is a wiki-style platform that enables easy collaborative creation of multilingual content in many non-English Wikipedias, by leveraging therelatively larger and more stable content in the English Wikipedia.

This project targets to create a parallel corpora for the improvement of statistical machine translations. Parallel corpara are traditionally created by professionals and are available only in a few languages of the world. This project is currently unfinished.

2.3.4. Online dictionaries

There are many online dictionaries. However with the introduction of Web 2.0, dictionaries also evolved. People have chance to enter words which are not on the dictionary. This way dictionary databases have been more accurate than before. http://www.seslisozluk.com is one of the best examples of online dictionaries that we can make use of.

3. TURC: TRANSLATION GAME

The goal of this work is to design a cooperative two player game for improving text that is machine translated from English to Turkish. For this purpose we designed a game called Translation with UseR Computation(TURC) and implemented a prototype of its core functionality. This game aims to serve as a proof of concept for a game based translation assistance approach. Experiences related to the deployment of the prototype are discussed in Chapter 5. In this chapter we describe the design of TURC.

3.1. Requirements

TURC aims to gather human input to be used in conjunction with machine language translation towards enabling a wider global access of Web content. The requirements for TURC are identified as follows:

- R1: Accessibility – In order to gather heterogeneous input the game must be accessible in terms of:
  - Availability – the game must be always available to play
  - Usability – ease of use must be high
  - Technically – computing resource requirements must be low
- R2: Source Material – A wide variety of interesting English content to be translated must be provided.
- R3: Machine Language Translator – A machine language translator from English to Turkish must be available.
- R4 : Players – When users enter the system they are considered as *player*s.
  - Identification – All users who enter the system must have a login id.
  - Real player – A currently logged in player who is available to play.
  - Virtual player – A simulated player, which is constructed from a previously recorded game.
  - Score – The scores accumulated from games played must be kept.
  - History – All user game information must be recorded.

- R5: Game play – A synchronous game play, where two players are matched up for improving the same piece of text is required:
  - Player pool – All users who are logged in must be kept in a player pool, which keeps track of all active players.
  - Matching – When players indicate they want to play a game, whenever possible, they must be matched with a real player. In the case when there are no real players available, they must be matched with a virtual player.
  - Synchronous – All games are played synchronously, where the time is recorded.
  - Source text – The English text that is to be translated must be visible.
  - Target text – Initially the source text which is machine translated to Turkish must be presented. Thereafter, the text must show its current state which is the result of the player's corrective actions.
  - Actions – Actions to correct the target text must available. The player must be able to *add*, *delete*, *modify*, and *move* words within the text.
  - Score – The score associated with the game must be visible.
- R6: Scoring – A scoring mechanism for evaluating two sets of translation corrections is required. This mechanism must award higher score based on similarity in order to motivate agreement between matched players. It must also take measures to determine cheating in the cases where players may try to simply achieve agreement instead of actually improving translation.
- R7: Responsiveness – The game must be responsive. It must match up a player with another, respond game actions, calculate awarded points within a reasonable time.
- R8: Persistency – Player actions and scores must be persistent over game sessions.
- R9: Motivation – The game must be fun to play. For this reason motivating features such as earning points and achieving a place in a visible high-score lists are required.

3.2. The Game Model

We propose a cooperative game model for improving translations. The model is explained in three parts:

- *The flow*, which describes how the game will be played
- *Player actions*, which describe the actions needed for players to specify improvements
- *Scoring*, which introduced a method for scoring the proposed improvements by a matched pair of players.

3.2.1. Game Flow

The game TURC must always be available to play. People who want to play the game must register so that their activities can be preserved for future use. Lets define the following terminology:

- Available player: A player in the player pool who has *available* status at current time.
- Bot player: A robot that acts like a player, using previously played game history
- Countdown: A timer that starts from a predetermined value
- Drop: The situation when a player loses connection to a game session
- Game Session: A game piece through the first sentence of an article to its end
- Match: A pair of players who are matched up in a game session
- Player: A user of TURC game
- Player pool: A set of players who are logged in at current time.
- Pool flush: Matching players in the pool that has available status for each predetermined time interval
- Score: A numeric value assinged to an improvement submitted by a pair of players.
- Source text: Text to be translated.
- Submit: A player action for submitting their correction for evaluation.
- Translated text: The text that is the translation of the source text.

The game has a pool of players, where it randomly matches two players who doesn't have any communication in between. In the case of an odd number of players, a bot player is generated using previously recorded player actions to make the number

```
Input: Set of tasks and processors
Output: Mapping of tasks to processors
while game server is up and running do
    if Player p logins then
        CreateAvailablePlayer(p)
    foreach Pool flush do
        foreach Available player i do
            if Player pool has Available players then
                Match(i,RandomAvailablePlayerInPool())
            else
                Match(i,GenerateBotPlayer())
    foreach Match j do
        GetSourceText()
        Sentence = ParseSourceText()
        foreach Sentence s do
            while !j.Player1.Submit and !j.Player2.Submit and Countdown¿0 do
                Translate(s)
                j.Player1 makes actions j1
                j.Player2 makes actions j2
                if j.Player1 or j.Player2 submits then
                    initializeCountdown()
            CalculateScore(j1,j2)
            StorePlayerActions(j1,j2)
            if j.Player1 drops or j.Player2 drops then
                GenerateBotPlayer()
        CreateAvailablePlayer(j.Player1)
        CreateAvailablePlayer(j.Player2)
```

Figure 3.1. TURC Game Flow Algorithm

of players in the pool even. Thus each player is matched with another player and the games begin.

In the game, each pair of player is presented the same English text and the Turkish translation of its sentences in sequence. For each sentence players has a set of actions aiming to improve the quality of the translated text. When a player decides that no more actions are necessary, he submits his modified sentence. After the submition of one of the players a countdown initializes, during which either the other player submits or a timeout occures. Than the next sentence will be shown. The process between the appearance of the initial sentence to the appearance of the next sentence is called a **game turn**.

In the end of each turn, a score is generated according to the correspondance of player actions. If for some reason a player leaves the game after the game has started, scoring is determined versus a prerecorded set of actions from the history of another player that already played the same text.

Game turns are played until the end of the main text. When all the sentences of the main text are finished, both players are returned to the pool. Eventually this forms a continual loop, again starting with the matching of random players.

3.2.2. Player Actions

Actions can be done while editing a text can be grouped in four topics:

- Adding a new word – This action is necessary whenever the translated text lacks the words needed for the translation. This action will be referred as actAdd(word) further on, “word” representing the word to be added.
- Deleting an existing word – The action to remove the word from translation. There can be three possible ways:
  - Wrong translation of a word occures when the machine translation has made a mistake. This action will be referred as actDel!(word) further on, “word”

Input: Set of tasks and processors

Output: Mapping of tasks to processors

if *actDel(word)* **then**
| InsertIntoAddedWords(word)

if *actDel(word)* **then**
| InsertIntoDeletedWords(word)

if *actEdit(old,new)* **then**
| InsertIntoAddedWords(new)
| InsertIntoDeletedWords(old)

if *actMove(prev,word,next)* **then**
| InsertIntoMovedWords(prev,word,next)

Figure 3.2. TURC Player Actions

representing the word to be deleted.

- Untranslated word occures when the machine translator doesn't translate a word. This action will be referred as actDel?(word) further on, "word" representing the word to be deleted.
- Unnecessary word occures when the machine translation has unnecessary words that are not needed in the translated sentence. This action will be referred as actDelX(word) further on, "word" representing the word to be deleted.

All three methods will be referred as actDel*(word).

- Editing an existing word – This action is used to edit a word when its extension needs to be changed. This action will be referred as actEdit(old,new) further on. "old" represens the word to be edited and "new" represents the final version of the word.
- Changing the order of the words – This action is used when there is a mistake in the sequence of words in the translated sentence. This action will be referred as actMove(prev,word,next) further on. "prev" and "next" are representing the previous and the next word in sentence order respectively. "word" is representing the word that is moved.

3.2.3. Scoring Mechanism

Scoring mechanism is designed to produce points whenever user actions corresponds in a game turn. These points are given to users regarding the changes of the translated text.

Points are given only to the mutually agreed changes. In order to gain points players have to do similar actions. Since the two players cannot communicate and know nothing about each other, the easiest way for both to produce the same output is by improving the translation.

Different correspondence will generate different points. Points are classified as high, medium and low.

There are four possibly actions to gain points:

- actAdd(word): If both players use this action to add the identical word, they'll be awarded high points. However, if each of the players adds the same word with different extensions or a synonym, than rather than gaining no points, they should be awarded low points.
- actDel*(word): If both players use the delete action for the same word, they will gain medium points. If also their classification are same they will get aditional low points.
- actEdit(old,new): If this action is corresponding in both players, the points of actDelX(old) and actAdd(new) are calculated and their sum is the total point of this action.
- actMove(prev,word,next): This move should give points only at the time of submittal. If a words place is changed in both players to the same place, making sure its different from the initial machine translation, players gain high points.

A fifth option is available for prerecorded gameplay. If your previous game is selected for the opponent of a new singleplayer game, the points their actions get will

Input: Set of tasks and processors

Output: Mapping of tasks to processors

foreach *word w in AddedWords* **do**

    if *PartnerAddedWords.contains(w)* **then**
        Give points!

foreach *word w in DeletedWords* **do**

    if *PartnerDeletedWords.contains(w)* **then**
        Give points!

foreach *word w in MovedWords* **do**

    if *PartnerMovedWordsWithNext.contains(w+w.next) and !OriginalSentence.contains(w+w.next)* **then**
        Give points!

Figure 3.3. TURC Score Algorithms

also be given to you, increasing your overall score.

3.2.4. Trust

Trust is a relationship of reliance. A trusted party is presumed to seek to fulfill policies, ethical codes, law and their previous promises [27]. In our game scenario, trust issue identifies whether players make reasonable improvements, try cheating or not.

3.2.4.1. Cheating. Cheating is defined as pretending to obey the rules of the game but secretly subver them to gain advantage over another player [28], in our case gaining more points.

The primary subject should be randomness. Paired players must be unknown to each other. If they would have any other means of communication, then they can agree on making same actions, which indeed won't improve the translation.

A large community of players may have agreed on doing the same actions outside of the game by using other social web applications, such as forums. An example of this can be agreeing on deleting all the words that appear in the sentence. Such methods can be overcome by a backtranslation algorithm. A backtranslation algorithm translates the submitted text back to English and looks for the correspondence with the original English sentence using a machine translation evaluation metric. The similarity should be above a certain threshold value.

Other trivial methods such as adding the same word after and after also can be checked in code level.

3.2.4.2. Player Levels. If it is ensured that players cannot cheat, a player that can be trusted can be determined using total score values. Players can be ranked according to their point intervals. People with high points are the ones that spent a long time to play the game and have their actions corresponde with at least one other player.

In prerecorded gameplay mode, sorting out non-played or untrusted user actions is a good improvement. This way chances of improvement correspondences are increased.

3.3. Discussion

3.3.1. How to make a game enjoyable?

These are the methods that can make a game enjoyable:

Score Keeping: One of direct methods for motivation is assigning points in a success. It clears the objectives and winning conditions. Also a score summary will provide performance feedback to the players [16].

Player Skill Levels: Player skill levels, “ranks”, are another way to incorporate motivation into a game. Based on the number of points they have, players would have new ranks and advancing to a new rank would be a direct motivation. This can be

implemented proportionally with the score of the player. On the other hand it may be designed as the implication of how many games a player has played.

High Score Lists: Another method for motivation is high score lists. The score needed by players to be listed on a high score list varies in terms of difficulty relative to the lists time period, ranging from the past hour or week to all the way to the history of the game [16].

Timed Response: Setting time limits for game sessions introduces challenge into a game in the form of timed response [16]. Players are told to complete a designated number of problem instances within a limited time period. If they accomplish it, they may be given extra points for their performance. This is expected to keep the game challenging because it makes the game less straight forward. However this can reduce the quality of output also. Drawbacks should be examined carefully.

Randomness: Players should be paired randomly in order to avoid cheating as well as inputs or a particular game session, which makes the difficulty vary, thus keeping the game interesting and engaging.

Area of Interest: In order to keep attention of the players, the text to be translated should be something interesting.

3.3.2. How to make a game productive?

In order to be productive, the game should supply accurate outputs. Output accuracy means the quality of the output. System's output accuracy should be maximized.

Random matching: In a large scale, players paired randomly have no way of knowing their partner's identity so have no easy way to agree ahead of time on any cheating strategy. Thus, under random matching the probability of two or more cheaters using the same strategy being paired together should be low.

Player testing: Randomly presenting already known information to the players, or else presenting fake information and testing the players inputs would supply a trust value for the player. If players answer doesn't match with the known answer the player will be marked as suspicious and his answers will be less trusted.

Repetition: The game is designed such that it will consider an output correct if it occures too many times repetitively. The output will be considered correct until a certain number of players have entered it. This strategy for determining correctness enables to guarantee correct output with arbitrarily high probabilitiy.

4. PROTOTYPE

We have developed a web application to test our model. Our prototype gives a source text and its translation to two players, who specify a set of improvements to the translation. The suggested improvements of the players are evaluated to produce a score. This chapter describes this prototype in detail.

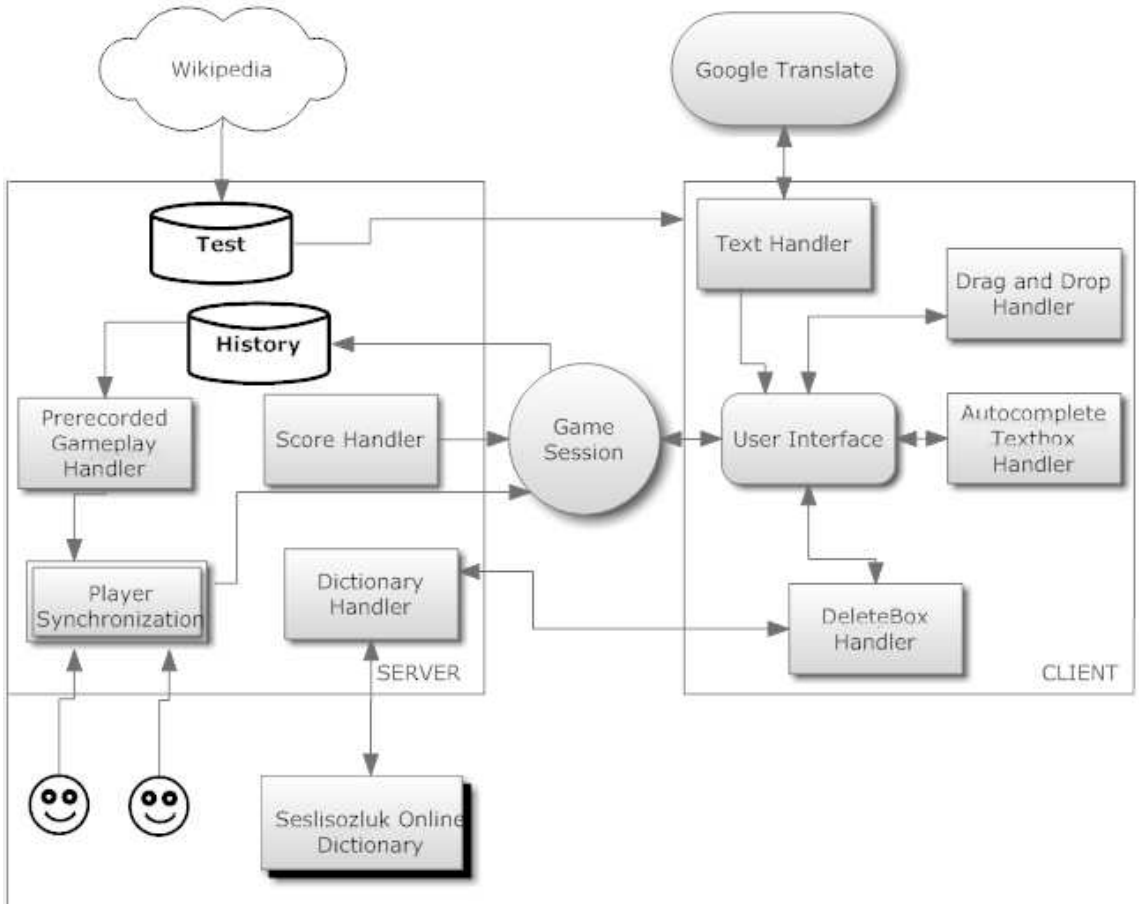


Figure 4.1. Prototype Overview

4.1. Software Environment

In the development of our prototype we tried to take being open source and platform independent into consideration when choosing technologies. This is for the ease of further development of our system. For this purpose we have selected Java as

our programming language. On the other hand AJAX (shorthand for asynchronous JavaScript and XML) is widely used for web development.

We have used Google Web Toolkit (GWT) for building our website. It is an open source set of tools that allows web developers to create and maintain complex JavaScript front-end applications in Java.

We used Eclipse as software development kit, because of its multiple platform support as well as large number of plug-ins that supports the tool, including the GWT.

We have used Jetty server, which will be described in later sections, on Linux environment.

We have used MySQL database. Because of its open source nature and quality as a relational database.

We have used various browsers, including Microsoft Internet Explorer, Mozilla Firefox, Opera and Google Chrome for testing purposes.

4.2. System Source

4.2.1. Text to be translated

The game needs English text to be translated into Turkish. Wikipedia is selected for this issue. Wikipedia has over 3,000,000 English , on the other hand just around 200,000 Turkish articles [2]. After the testing stage, the project can collect data using all the articles by continuing to make users translate. This opportunity is found promising.

Also, one of the main reasons selecting Wikipedia is its semantic availability in DBpedia project [29]. This way after the testing of the model the project can come to life and easily continue the translation operation through all the Wikipedia content by

taking the required text online. In addition it has additional properties such as photo collection of some articles, which indeed can be useful for the attraction of users and can increase fun of the gameplay. Also our output may be an input for DBpedia.

Wikipedia is generated and edited everyday by a community of users, therefore its context is up to date. In terms of entertainment we have selected a test set of 100 among most clicked 300 Wikipedia articles of 2009 [30]. These contain different subjects from popular TV Series to the famous ship Titanic.

We took the abstracts of the articles, which are mainly summarization of the complete text. The length of the abstracts differs from 2 to 10+ sentences. The text is automaticly divided into sentences with respect to the full stop. Before that as a preprocessing, dots in the abbreviations like "U.S. , Bros. , Dr. , St. , No. , INC. , Jr. , Mr. , Mrs. , Vol. , etc. . . " are discarded.

4.2.2. Translator

Any machine translation mechanism can be used with the project. We have choosen Google Translate for several reasons. First of all its easily reachable and it has an inbound mechanism to improve itself with suggestions. It has one of the few translators that supports English-Turkish translation which was launched online at 12^{th} stage of the program in January 30, 2009. Currently it has 51 different languages, which means we can create a version of the game for each of these languages by a minor tweak.

The decisive factor to choose Google Translator was its statistical machine translation approach [3]. Unlike other translation services using SYSTRAN and rule based approaches, Google implemented a system where translations are generated on the basis of statistical models the parameters of which are derived from the analysis of bilingual text corpora (parallel collection of texts). There is not enough amount of English-Turkish bilingual data, so the translation quality is not so good.

These systems can be improved by supplying more bilingual text or backfeeding the corrected output to the system. Google Translate has its suggest a better translation option available to the public. By using this option, users correct the wrongly translated sentence and submits a better translation.

4.3. User Interface

The game is designed to be a web page. It is hosted in "http://www.turcbeta.com/" The first impressions when a user visits the webpage is a blueish menu. This menu is consisted of four tabs , "How to play?" , "PLAY", "HI-SCORES" and "About" respectively.

4.3.1. How to play?

This tab is automaticaly focused when the page is first loaded. It describes how the game works and what the users should do. It has a testing place where all the options available ingame are described and can be tested for familiarity. This section also explains incentives to play the game, as well as giving hints and tips to the user.



Figure 4.2. A screenshot from How to play? screen

4.3.2. PLAY

This tab initially consists a combo login/register panel. People tend to quit this kind of website because they don't want to deal with the registration process. For this reason as a design decision, we implemented a combo login/register system, where if a user enters a registered username with its password can login the system, while a non-registered username and password creates an account for the user. This way we intended an ease in registration.

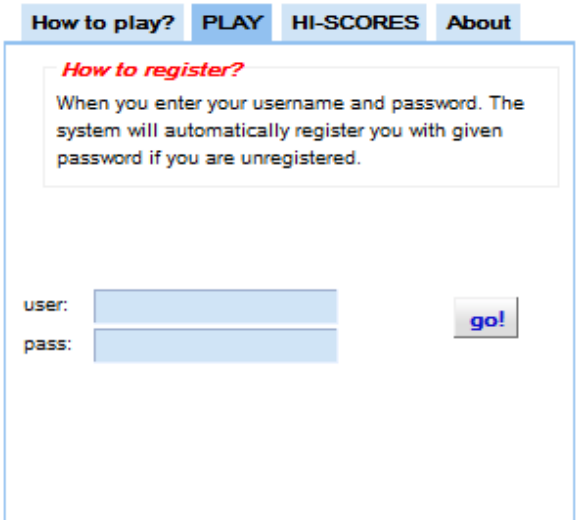


Figure 4.3. A screenshot from PLAY tab before login

When the user successfully completes the login/registration process, the context of the panel becomes the information about the user with a "start a new game" button. If the user presses the button, a counter from 30 starts to countdown, looking for other players to start a multiplayer game. If 30 second limit passes than a single player game is generated. Time limit is selected for half a minute to increase the number of multiplayer games.

Game screen appears when a game is initialized, which is described later in "Game Interface" section.

4.3.3. HI-SCORES

This section lists the users with the highest points. Through our two week testing only all the time hi-score list were available. However, currently daily and weekly lists are introduced. This makes the game challenging both for old and new users.



Figure 4.4. A screenshot from Hi-Score screen

4.3.4. About

In this part a brief description of the project is given along with the contact information. Users are asked to e-mail bug reports and feedbacks.

4.4. Game Interface

The user interface of the gameplay is divided into five parts:

4.4.1. North Part

In the north part, whole article to be translated is given. For testing purposes, the system takes the article from our database. However, in the future it can be selected on the run.

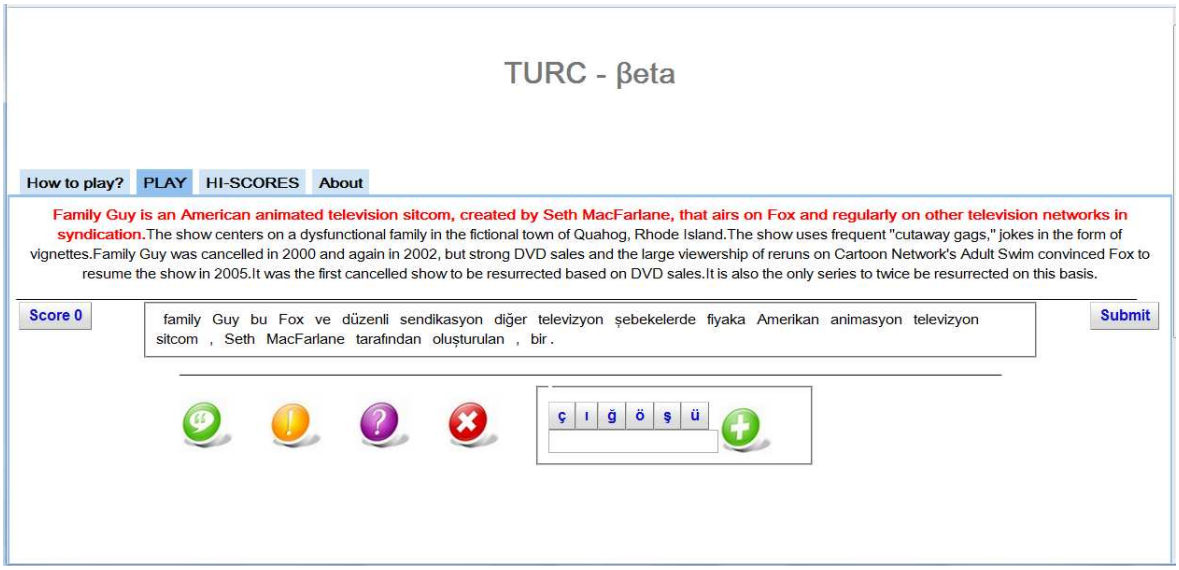


Figure 4.5. A screenshot of the game interface

The current sentence is marked with red to make it more clear.

4.4.2. Center Part

In the center part, the Turkish translation of the marked sentence is given. In this section every word has drag and drop availability. User can change the order of the words, can edit and delete the words using the tools in the south part. User finalizes improving this sentence before submitting.

4.4.3. West Part

In this part there is the score counter. Scoring is calculated according to both the players' actions. The algorithm will be described later.

4.4.4. East Part

This part is a simple submit button. Submit button acts as an informative counter in multiplayer game. In the screen of the first submitted player, it states the condition

of waiting the opponent. On the other screen it warns the player that it has limited time left. This is currently set to ten seconds. When the time is up it auto-submits and passes to the game to the next sentence.

4.4.5. South Part

In the south part of the user interface, there are tools to add, edit or delete the words in the translated sentence. There are four adjacent boxes in the left part, and a add/edit panel containing several items including a textbox on the right side.



Figure 4.6. a. Edit b.Wrong Translation c. Untranslated d. Doesn't Needed e.Add

Edit Box: This is used for editing. When a word is dragged and dropped inside of this box it will remove the word from the interface and put it in the editable textbox of the right side. This can be manipulated and added to the interface later on.

Wrong Translation Delete Box: This box is for wrong translation of words. Users are encouraged to drop wrong translated words inside this box, which will eventually remove the word from the interface and searches it through dictionary. Opens a dictionary panel and shows the results.

Untranslated Word Delete Box: Users are encouraged to drop untranslated words they wish to delete into this box. It will remove the word from the interface, search it through dictionary and display the results by opening the dictionary panel.

Doesn't Needed Delete Box: This is a simple delete box. It will erase whatever you drop in it and doesn't respond in any other way. This box is for the unclassified words, that are not wrong translations neither untranslated nor needs an edit. They just shouldn't be in the sentence.

Add/Edit Panel: This panel has a textbox, which can be used to alter text, a box shaped button to add the word to the interface and Turkish characters which are not included in a standard English keyboard. After either a word drop in edit box, or writing a word from scratch hitting enter button, or clicking the box inside the panel will add the word to the interace, at the end of the sentence. Turkish characters are for the ease of the users which don't have that characters in their keyboards.

4.4.6. Dictionary helper

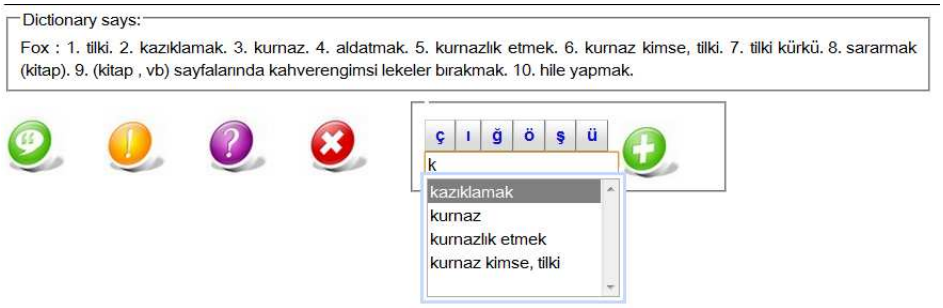


Figure 4.7. Dictionary helper and autocomplete textbox in action

When a word is deleted for wrong translation or being not translated, a dictionary helper menu appears in the south part. This menu gets the information by querying the word from http://www.seslisozluk.com, an online dictionary. If the word is in Turkish, than it is translated back to English and the result is translated in the online dictionary.

All the results are added to the textbox in the Add/Edit panel as auto completion suggestions.

4.5. Server Client Integration

- q0: This is the initial state, when a player initializes a game a 30 second count-down starts, trying to match the player with another.
- q1: This is the single player game state. P_1 plays the game with a bot.
- q2: This is the two player initial game state. Each player are shown the text, the

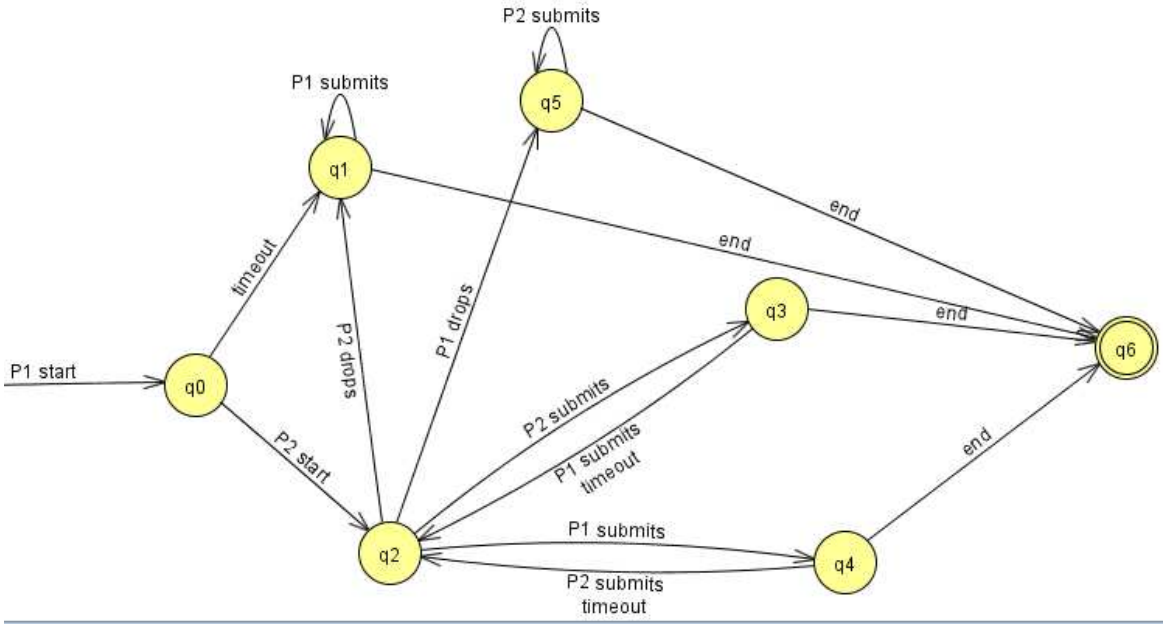


Figure 4.8. A diagram representing the finite state machine of the prototype

sentence and actions can be made.

- q3: This is the state when P_2 submitted. A 10 seconds countdown starts, and a response from P_1 is waited.
- q4: This is the state when P_1 submitted. A 10 seconds countdown starts, and a response from P_2 is waited.
- q5: If P_1 drops than P_2 continues its game as a single player with a bot.
- q6: This is the end state when the article is finished.
- Timeout: A timeout that occures when a countdown reaches 0.
- Drop: If a player either loses connection or leaves the game then that player is dropped from the game.
- Submit: Player indicates that he has finished his actions during this turn.
- End: This is the end of article as well as the game session, because all sentences are translated.

4.5.1. Multiplayer Game Infrastructure

The game is developed to reach large number of players and it is intended to be played at the same time by the users. Thus game infrastructure must support high volume of users.

In a single-player game, like the majority of other web applications, connections between the server and client can be closed during the frequent pauses in the system, for example while the players are using with their interface which is a client side application.

However a multi-player game has very different traffic profile to such traditional web applications. While a user is playing, it also sends requests to the server to asynchronously obtain the status of the other player. Thus a multi-player game almost continuously needs a connection and it means if you want thousands of users you will need thousands of connections, if you want tens of thousands of users, then you need tens of thousands of simultaneous TCP/IP connections. This is a challenge for java web containers. This may be avoided using NIO libraries, delivering asynchronous events to the users by the server.

However AJAX technology doesn't allow such a structure. In AJAX, clients poll the server for events, because servers are unable to deliver asynchronous events to the clients. This may cause a busy polling loop, which is often avoided by holding onto a poll request until either there is an event or a timeout occurs. This structure will cause an idle player to have an outstanding request waiting on the server which can be used to send a response to the client the instant an asynchronous event occurs.

The solution used to overcome such issues is using a Jetty server. A new feature is introduced in Jetty 6, which is called "Continuations" [31]. It simply creates a Continuation object which is used to suspend the request and free the current thread. If the resume method called or a timeout occured, the request is resumed.

Jetty Continuation offers the possibility of suspending a thread and then resuming it later. The first time the request handler suspends the continuation, an exception is thrown, which propagates out of all the request handling code and is caught by Jetty and handled specially. Instead of producing an error response, Jetty places the request on a timeout queue and returns the thread to the thread pool. When the timeout expires or the resume function of the continuation has been called, than the request is taken from the queue and is retried. This time when the suspend method is called, either the event is returned or null is returned to indicate a timeout. The request handler then produces a response as it normally would. This way, required minimum thread number is greatly reduced and it is possible to host the game for large number of users.

4.5.2. Player Synchronization

One of the most challenging aspects of developing a multi-player game is to synchronize the players. They have to interact with each other from the start until the end of the game through the server. For this reason we have implemented a first come first served structure.

When a player initializes a game, he will be immediately matched with a player currently waiting for a game to start. If such player don't exist, a game id is given and a waiting countdown for 30 seconds starts. If another player starts a game before the countdown reaches 0 they are matched, else a prerecorded game sequence initializes as a single-player game session.

4.5.3. Single-player System

At the beginning of a single-player game session, a random article is selected among previously played games by other players, excluding the games this player has already played. Here only games that players over 5000 points played are taken into consideration to supply a more trusted set of actions. According to our tests more than two well played game sessions can take you up to 5000 points at minimum.

Each sentence of this article is presented to the player sequentially after being translated, asking to make actions to improve the translated text until the end of the article. For each sentence a random set of actions from the history of other users is selected to calculate points.

4.5.4. Two-player System

In a two player session, the first player that initialized the game, is considered the host. A random article is selected among the articles host hasn't played yet.

Each sentence of this article is presented to both players sequentially after being translated, asking to make actions to improve the translated text until the end of the article. When a player finishes the actions he will make, he submits. After the submittal of the other player 10 seconds is given to the second player. Note that in our game we didn't set a predetermined time limit, but we set relative time limit. This is due to the variance of translation lengths of different sentences. At the end of the 10 seconds, the sentence is auto-submitted. If any of the players quit the game or lose connection, than immediately the session is transformed to a single player game session.

4.6. Score System

Scoring is calculated for both players and the sum is considered as the total score for each player for a particular game turn.

4.6.1. Adding a new word

This action is rewarded with 25 points each. Added word can actually be different. If the word is identical than a total of 50 points effects the actual score. If the words are different by only an extension a total of 25 points are awarded.

The algorithm searchs word1 in word2 and vice versa. If found 25 points are

given. As an example, lets assume “kelimeler” and “kelimeleri” are added by the first and second player respectively. Then “kelimeler” can be found in “kelimeleri”, but the opposite is not true. In such case only 25 points are given, however if both player added just “kelimeler” than they will be awarded a total of 50 points, 25 for each match. This difference is the result of agreement.

4.6.2. Deleting a word

The words in the given sentence are known. Among these words each user has an option to delete the word from the sentence to be submitted. There are three different boxes for delete operation. If the same words are deleted, 15 points are given for each. If both the word and the box used are the same an additional 10 points are given for each of the players. This way players are encouraged to classify the words by using deletion boxes.

4.6.3. Editing a word

Editing a word works as a mixture of adding a word and deleting a word. However the word isn’t compared with the words that have been droped in the deletion boxes.

4.6.4. Order of the words

Players has the ability to change the order of the words. According to our model, if both players make the same changes, they’ll get points. This is determined by checking word pairs. Each sequential pair, that is not sequential in the original text, but in both players is awarded 20 points. Consider a sentence such as:

“word1 word2 word3 word4 word5 word6.”

The output sentence of the first player:

“word1 word4 word5 word2 word3 (word6+’dir’).”

The output sentence of the second player:

"word2 word1 word4 word5 word3 word6."

Here we cross check the correspondences of pairs. Starting from the output sentence of P_1: word1+word4 exist in P_1, but doesn't exist in the original text. We check it with P_2 and we see that it is there. So 20 points are given and the case is written in to the database which will be later described.

We continue our check with word4+word5. Because this pair exists in the original sentence it is ignored. word5+word2 isn't in the P_2's output so it is also ignored. word2+word3 is again in the original sentence.

Here in word3+(word6+'dir'), word6 is edited and an extension is added. Because of this edit, it is not available in P_2, however word3+word6 exists in word3+(word6+'dir'). So partial points are awarded. These words are also checked with swapping places, so if the extension is in the first word than it still will be detected.

4.6.5. Prerecorded Game Points

Players can be disturbed to have a bad partner, not gaining any points as they make certain improvements to the translation. To avoid this frustration, if your actions in one of your previous game is used in a prerecorded game session, you will gain points as if you were playing in that session. Thus you'll be given other chances to earn points.

4.7. System Output

Currently our prototype stores player behaviour and statistics. Each added, deleted, edited, moved word is written into our database.

Our database has five tables, articles, players, sentences, test and words.

Articles table stores a user's played articles. It has the identifiers of the user, the game, and the article with total number of completed sentences, total time spent, total score gained and on which date.

Players table stores user information such as username and password, total score, best score, date of register, date of last login, total time of play, number of started and number of finished articles.

Sentence table is a rather large table with the identifiers of the players in the game session, the game, the article and sentence with information of added and deleted words for each possible box, the scores gained by each, original sentence as well as the submitted, the game information of being single or multi-player, timespent for this sentence and lastly date.

Test table is consisted of 100 Wikipedia articles with the information regarding their name, abstract and id.

Words table has every single added, deleted or moved word. It has the word used in an action, its previous and next words in the submitted sentence, the user that did the action, the identifiers of article, sentence, game and the action with the exact date.

5. Experiment and Results

In this chapter the experiment over our prototype is being discussed, evaluation metrics are introduced and results are presented.

5.1. Setup

The experiment is done through a publicly accesible web site. Our prototype has been made operational on 25^{th} of August. Announcements through mailing lists and social channels such as facebook are made.

5.2. Evaluation Metrics

To evaluate our system, we have to define metrics the metrics used in evaluation of a GWAP. Normally efficiency is a natural metric of evaluation. There are many possible ways of implementation, so some are more efficient than others. In order to find the better one a set of metrics is needed.

Efficiency of standard algorithms is by counting atomic steps. For instance Quicksort is said to run in O(n log n) time, which means it needs approximately n log n computational steps to sort a given list. In the case of GWAPs computational steps are unclear. Therefore efficiency is defined by other means such as, throughput, enjoyability and expected contribution [16].

5.2.1. Throughput

Throughput of a GWAP is defined as the average number of problem instances solved, or input-output mappings performed, per human-hour [16]. As an example, throughput of the ESP Game is roughly 233 labels per human-hour [16]. This is calculated by examining how many player inputs, or images, are matched with outputs, or labels, over a certain period of time.

Here learning curves and variations in player skill must be considered in calculating. In general, games has a certain type of learning, meaning with repeated game sessions over time, players become more familiar with the game thus more skilled at the game. Learning can result a faster game play over time. To take learning curves and changes in player speed over time as a result of learning, throughput is defined as the average number of problem instances solved per human-hour.

The higher the throughput is, the better the game. But throughput is not enough by itself. Because how many problem instances solved doesn't mean anything if nobody wants to play. Therefore a combination with enjoyability is needed.

5.2.2. Enjoyability

Enjoyability is a relative term, and this makes it difficult to quantify. Enjoyability depends on the precise implementation and design. Even a tiny modification of the user interface or scoring system can significantly affect how enjoyable it is to play. For this reason, to measure enjoyability, "Average Lifetime Play" (ALP) is taken into account. ALP is the overall amount of time the game is played by each player averaged across all people who have played it.

$$ALP = \frac{\text{Overall amount of the time the game is played}}{\text{The number of players}} \tag{5.1}$$

ESP Game plays for a total of 91 minutes on the average [16].

5.2.3. Expected Contribution

Expected contribution is the combination of average problems solved per human hour (throughput) and expected time for a player to spend in the game (ALP). Thus expected contribution indicates the average number of problem instances a single hu-

man player can be expected to solve by playing a particular game [16].

$$ExpectedContribution = (Throughput) * (ALP) \tag{5.2}$$

This approach is a fairly stable measure of a game's usefulness. Its only weakness it the disability to capture certain aspects of games such as popularity and contagion.

5.3. Results

5.3.1. Statistics

Table 5.1. Number of player agreements on improvements over two weeks

	Added	**Edited**	**Wrong translated**	**Untranslated**	**Directly Deleted**	**Moved**	**Total**
Matches	421	305	12	4	972	110	1824
Distinct matches	146	132	9	4	145	61	497

A total of 130 users registered the game over a two week period. This is approximately 9 registered users per day. Among them 99 submitted a sentence, on the other hand 31 of them just had a look. A total of 57 users has actually played and gained points. 32 of them returned to the website after their first game. 31 spent more than half an hour playing the game and 21 spent more than an hour.

An overall of 230 games played, which consisted the translation of 14 articles. A total of 132728 seconds has been spent playing the game which corresponds to 2212 minutes, which is 37 human hours. We have collected data for 770 English sentence. Players had agreement on 1824 improvements. Details can be seen in Table 5.1.

5.3.2. Evaluation Metrics

We had 37 human hours spent in our prototype, matching on 1824 cases. Thus we have a throughput of 49 cases per human hour. This is not close to the ESP game, however we must consider that, making translations is more time consuming than generating tags. It is better to compare with Verbosity game, which is a verbal process. Although we don't have their throughput value it is stated that on average, each player contributed 29.47 facts also played for an average of 23.58 minutes in one sitting. [18] We have 32 matches per user (considering the users that actually played the game), which is a little bit better than Verbosity.

We have observed an average lifetime of 38 minutes. Which means on the average a player spent 38 minutes in our game. Our expected contribution is 20.26 improvements per player.

5.3.3. Example Results

An example sentence:

"The RMS Titanic was an Olympic-class passenger liner owned by the White Star Line and built at the Harland and Wolff shipyard in Belfast, United Kingdom. "

and its translation is given as:

"Rms Titanic bir Olimpiyat sınıf yolcu uçağı , White Star Line ait ve Harland ve Belfast , Birleşik Krallık , Wolff tersane inşa oldu"

Here are what users did:

User1: "Rms Titanic bir Olimpiyat sınıf yolcu gemisi , White Star Line'a aittir ve Birleşik Krallık , Belfast'taki Harland ve Wolff tersanesinde inşa oldu"

Added words: gemisi Line'a tersanesinde Belfast'taki aittir

Edited words: Line tersane Belfast ait

Deleted as wrong translation: uçağı

User2: "rms Titanic White Star Line'a ait , Belfast Birleşik Krallık'taki Harland ve Wolff tersanesinde inşa olmuş , olimpik sınıfından bir yolcu gemisidir"

Added words: gem gemisi Line'a tersanesinde olmuş Krallık'taki olimpik sınıfından gemisidir ,

Edited words: gemisi Line tersane oldu Krallık sınıf

Deleted as wrong translation: Olimpiyat

Directly deleted: uçağı gem , , ve

User3: " White Star Line'a ait RMS Titanic , Belfast - Büyük Britanya'daki Harland ve Wolff tersanesinde inşa edilmiş Olimpik bir sınıf yolcu gemisi"

Added words: Line'a RMS - Britanya'daki gemisi edilmiş

Edited words: Line rms Britanya Liner edildi

Directly deleted: ve

User4: "Rms Titanic bir Olimpiyat sınıf yolcu gemisi , White Star Line'a aittir ve Birleşik Krallık , Belfast'taki Harland ve Wolff , tersanesinde inşa olmuştur"

Added words: Line'a Belfast'taki tersanesinde gemisi aittir olmuştur

Edited words: Line Belfast tersane ait oldu

Deleted as wrong translation: uçağı

User5: " RMS Titanic , White Star Line'a ait olimpik sınıfı bir yolcu vapurudur ve Büyük Britanya , Belfast'taki Harland ve Wolff tersanesinde inşa edilmiştir" Added words: Liner'ıdır Line'a Britanya'daki edilmiştir RMS , vapurudur sınıfı olimpik Britanya Belfast'taki

Edited words: Liner Line Britanya edildi rms Liner'ıdır sınıf Olimpik

Directly deleted: Britanya'daki Belfast

We can clearly see that people tend to improve the translation in similar ways. Although the output is somewhat different, most of the added, edited and removed words are shared. As an example we can see that all of the five users edited the word "Line" and modified it as "Line'a". 3 of 5 added Belfast'taki etc...

Note that, we can easily eleminate the words that are both in added and one of the deleted section. Here we see that user3 unintentionally misspelled gemisidir as gem and deleted it afterwards. Because it appears both added and deleted sections its a ingame word, that doesn't exist in neither output nor input.

Another example:

Original sentence:

"His novel versions of existing songs, mixing black and white sounds, made him popularand controversialas did his uninhibited stage and television performances."

Translated sentence:

“Mevcut ṣarkı romanını sürümleri , siyah ve beyaz ses karıṣtırma , onu - ve tartıṣmalı - olarak görev yaptığı sınır tanımayan sahne ve televizyon gösterileri yaptı popüler hale “

User1: Mevcut ṣarkı sürümleri , siyah ve beyaz sesleri karıṣtırarak , onu popüler hale getirdi - tartıṣmalı olarak - görev yaptığı sınır tanımayan sahne ve televizyon performansları gibi

Added words: gösterileri performansları sesleri karıṣtırarak getirdi gibi

Edited words: ses karıṣtırma

Wrong translation: gösterileri yaptı

Direct deletion: romanını ve

User2: Mevcut ṣarkıların deiṣik versiyonları , siyah ve beyaz melodilerini harmanlaması , yaptığı sınır tanımayan sahne ve televizyon gösterileri O’nu popüler ve tartıṣmalı hale getirdi

Added words: ṣarkıların novel deiṣik versiyonları sound sesleri harmanlama melodilerini harmanlaması onu O’nu getirdi

Edited words: ses harmanlama

Wrong translation: ṣarkı novel bir and sound karıṣtırma

Direct deletion: romanını sürümleri sesleri - - onu onu görev yaptı olarak

User3: Onun mevcut ṣarkılara yaptığı yeni versiyonlar , eselerinde kara ve ak sesleri karıṣtırması , onu yaptığı sınır tanımayan sahne gösterileri ve televizyon performansları gibi,popüler ve tartıṣmalı hale getirdi

Added words: ṣarkıların Mevcut yeni versiyonlarını Onun mevcut ṣarkılara yaptığı versiyonlar ak kara sesleri karıṣtırması eselerinde , performansları gibi , getirdi

Edited words: ṣarkı mevcut mevcut ṣarkıların versiyonlarını ses karıṣtırma

Wrong translation: romanını

Direct deletion: sürümleri siyah beyaz - - görev olarak yaptı

User4: Siyah ve beyaz sesleri karıṣtıran , mevcut ṣarkıların yeni sürümleri , sınır tanımayan sahne ve televizyon gösterileri onu popüler - ve tartıṣmalı - hale getirdi

Added words: ṣarkıların yeni sesleri karıṣtırma karıṣtırarak getirdi Mevcut karıṣtıran Siyah mevcut

Direct deletion: romanını ṣarkı ses karıṣtırma karıṣtırma mevcut karıṣtırarak siyah Mevcut olarak görev yaptığı yaptı

We can easily see the similarities in the added and deleted words in this example too. We can also say that every submitted sentence is an improved version of the original text.

5.4. Discussion and Future Work

Our results are as expected. We have proved that people add, delete, edit or move the same words in a sentence in a similar manner. A GWAP on language translation with our model can be really successful.

However, there are several reasons that hampered the success of our implementation of the game. First of all our testing period was very limited and it was summer, also ramadan, in which most of Turkish people tend to stay away from computers or not to do anything that involves thinking. These caused less people to play the game.

Even lesser two player game experiences occured (18 out of 230).

There were a couple of bugs through the testing stage, which may have caused problems to some players.

It is clear that Wikipedia articles are a bit difficult to translate for an intermediate English speaker. The sentence structures are usually too complex. There were some heavy words, which are not easily translatable. Also translating long sentences are sometimes boring. These articles are not easy to translate even for the experts.

We observed that our prototype is slow-paced which reduces some of the fun element. In our current implementation you have to wait until you submit the sentence to earn points. However, earning points immediately after an agreement may increase fun factor, also consentration of the user. Time constraints are note satisfactory. The two player 10 seconds is a bit short, if your partner presses too early.

We have seen that most of the players don't read the game manual. They try to learn the game with a trial and error approach. Because of this players don't use every deletion box, probably not knowing what will happen. Displaying the game rules during the countdown in the game initialization may overcome this issue.

We got several other feedbacks from our users. Socializing players with messaging,grouping etc. , ability to save and continue later, difficulty level of text to be given different levels of players, minigames to rate the output translations, improving user interface with article related images derived through semantic search are possible ways of and converting our game to a facebook application can be further improvements.

Also one final remark as a future work, we can use our own machine translator, so that we can feed our outputs to see the actual effects over the quality of the machine translations.

6. CONCLUSIONS

In this work, a model for improving and a prototype for generating data to improve machine translation between English to Turkish is developed. We have presented guidelines for designing a successful game with a purpose. We also uncovered methods that can be used in a translation game.

The prototype is meant to be played by two players simultaneously over the Web. Whenever there is odd number of players, a bot player is generated using actions from player histories in order to keep the game available all the time. In the beginning of a game session, an English source text is selected. This was chosen to be an article from Wikipedia for the prototype. Sequentially all the sentences from the selected article is being translated using machine translation. Players that don't have any communication, try to cooperate to improve the quality of the translation. As a result of successful cooperation, such as adding, deleting, editing or moving the same word in the similar manner, points are awarded to both players. Players challange to get a place in the hi-score lists by gaining more points. Also players can categorize deleted words as wrong translation, untranslated or not needed in translation. Whenever they delete a word marking with wrong translation or untranslated, a dictionary helper appears to be in service.

We have chosen to host only English to Turkish version, although our prototype supports 51 languages. The game is hosted on a publicly available web site. The experiment was started on 25^{th} of August. Results reported in the thesis are compiled from the data obtained 9^{th} of September. The evaluation is done by defined metrics such as throughput, enjoyability and expected contribution. Our expected contribution was found to be 31 improvements per player. Our game speed is observed to be slow-paced which hampered our throughput. However considering that this system is text based unlike other image based games, and translation is a bit more time consuming process, than other straightforward operations, this was expected. In order to improve the speed of the game an immediately responsive scoring mechanism is propositioned

as a future work.

Users have agreed on 1824 improvements. In which we observed that solid improvements are dense. Thus using human computation for improving machine translation is promising.

REFERENCES

1. O'Neill, E., B. Lavoie, and R. Bennett, "Trends in the Evolution of the Public Web", 2002, http://www.dlib.org/dlib/april03/lavoie/04lavoie.html.

2. Wikipedia, "Wikipedia, the free encyclopedia that anyone can edit", 2009, http://www.wikipedia.org/.

3. Google, "Google Translate", 2009, http://translate.google.com/.

4. Microsoft, "Bing Translate", 2009, http://www.microsofttranslator.com/.

5. Babylon, "Babylon Translation at a click", 2009, http://babylon.com/.

6. ESA, "Entertainment Software Association Webpage", 2009, http://www.theesa.com/.

7. von Ahn, L., "Games with a purpose", *Computer*, Vol. 39, No. 6, pp. 92–94, 2006, http://dx.doi.org/10.1109/MC.2006.196.

8. Wikipedia, "Human Computation Wikipedia article", 2009, http://en.wikipedia.org/wiki/Human-based_computation.

9. Johnston, V. S., "Method and apparatus for generating composites of human faces", 1992, http://www.google.com/patents?vid=5375195.

10. Digg, "Digg, discover the best of the Web", 2009, http://www.digg.com/.

11. Von Ahn, L., M. Blum, and J. Langford, "CAPTCHA: Using Hard AI Problems For Security", *In Proceedings of Eurocrypt*, Vol. 2656, pp. 294–311, 2003, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.13.3335.

12. von Ahn, L., B. Maurer, C. Mcmillen, D. Abraham, and M. Blum, "reCAPTCHA:

Human-Based Character Recognition via Web Security Measures", *Science*, pp. 1160379+, August 2008, http://dx.doi.org/10.1126/science.1160379.

13. 20Q, "20 Questions the neural net on the internet", 2009, http://www.20q.net/.

14. Hasbro, "Monopoly City Streets Game", 2009, http://blog.monopolycitystreets.com/2009/09/whats-going-on.html.

15. von Ahn, L. and L. Dabbish, "Labeling images with a computer game", *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 319–326, ACM, New York, NY, USA, 2004, http://dx.doi.org/10.1145/985692.985733.

16. von Ahn, L. and L. Dabbish, "Designing games with a purpose", *Commun. ACM*, Vol. 51, No. 8, pp. 58–67, August 2008, http://dx.doi.org/10.1145/1378704.1378719.

17. von Ahn, L., R. Liu, and M. Blum, "Peekaboom: a game for locating objects in images", *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pp. 55–64, ACM, New York, NY, USA, 2006, http://dx.doi.org/10.1145/1124772.1124782.

18. von Ahn, L., M. Kedia, and M. Blum, "Verbosity: a game for collecting common-sense facts", *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pp. 75–78, ACM, New York, NY, USA, 2006, http://dx.doi.org/10.1145/1124772.1124784.

19. Law, E. and L. von Ahn, "Input-agreement: a new mechanism for collecting data using human computation games", *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pp. 1197–1206, ACM, New York, NY, USA, 2009, http://dx.doi.org/10.1145/1518701.1518881.

20. Chamberlain, J., M. Poeso, and U. Krutchwitz, "A Demonstration Of Human

Computation Using The Phrase Detectives Annotation Game.", I-Semantics.

21. Foldit, "Fold it, Solve puzzles for science", 2009, http://fold.it/.

22. Wikipedia, "Fold it Wikipedia article", 2009, http://en.wikipedia.org/wiki/Foldit.

23. Wikipedia, "Google Translate Wikipedia article", 2009, http://en.wikipedia.org/wiki/Google_Translate.

24. Google, "Google Translator Toolkit", 2009, http://translate.google.com/support/toolkit/bin/answer.py?hl=en&answer=147809.

25. Koehn, P., "A Web-Based Interactive Computer Aided Translation Tool", 2009, http://www.caitra.org/.

26. Kumaran, A., K. Saravanan, N. Datha, B. Ashok, and V. Dendi, "WikiBABEL: A Wiki-style Platform for Creation of Parallel Data", 2009.

27. Wikipedia, "Trust in Social Sciences Wikipedia article", 2009, http://en.wikipedia.org/wiki/Trust_%28social_sciences%29.

28. Wikipedia, "Cheating in online games Wikipedia article", 2009, http://en.wikipedia.org/wiki/Cheating_in_online_games.

29. Auer, S., C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "DBpedia: A Nucleus for a Web of Open Data", pp. 722–735, 2008, http://dx.doi.org/10.1007/978-3-540-76298-0_52.

30. Wikistics, "Yearly page hits for en.wikipedia.org", 2009, http://wikistics.falsikon.de/2009/wikipedia/en/.

31. Foundation, C., "Jetty Server Documentation: Jetty Wiki", 2009, http://docs.codehaus.org/display/JETTY/Jetty+Wiki.