DODIOM RU: A GAMIFIED CROWDSOURCING CHATBOT FOR IDIOM CORPORA CONSTRUCTION IN THE CASE OF THE RUSSIAN LANGUAGE

Batuhan Duru Yeltekin

SABIS SUN International School
Baku, Azerbaijan
July 2022

Contents

List of Figures

List of Tables

FOREWORD

With this paper, we are expanding the purview of the initial research to the family of Slavic languages, in the hope that this game can be recreated in these languages. We hope that in the future, all Slavic countries will attain peace.

ABSTRACT

One of the most common setbacks of modern natural language processing (NLP) applications (e.g., Google Translate, Apple's Siri, and Samsung's Bixby) is their underdeveloped ability to correctly process multi-word expressions (MWEs). MWEs exist with both compositional and non-compositional/idiomatic meanings. Differentiating between the meanings in which an MWE has been expressed and interpreting them correctly has proved to be an arduous task for the aforementioned NLP applications, mainly due to a lack of high-quality data for language models to train on. To combat this issue caused by the scarcity of valuable data, Associate Professor Gülşen Eryiğit and master's candidate Ali Şentaş from the Faculty of Computer and Informatics at Istanbul Technical University developed a chatbot in the Telegram messaging app that utilizes a game-like structure and interface to crowdsource essential training data for such artificial intelligence models to train on. The chatbots, originally developed by ITU members, accept data entries in Turkish, English, and Italian. I was fortunate that the authors of the original article gave me the opportunity of working with Russian – a language that harbors a complex morphology and demands the involvement of a Russian linguist with a deep knowledge of Russian grammar structure, like my mother – and creating the Russian version of the chatbot. Since Russian is a popular language not only in the Russian Federation but also in post-Soviet countries, developing this chatbot would constitute a valuable contribution to the field of NLP and its real-world applications in Slavic-speaking countries. The focus of this research is to understand the morphological analysis of the Russian language and to introduce a crowdsourcing and crowd-rating approach, which is implemented and tested for the first time in the literature on idiom corpora construction. This paper also investigates one of the most effective finite-state transducers for the Russian language, UDAR. An NLP toolkit named Stanza was used in this project for its easy integration. Stanza helped effectively lemmatize MWEs over a period of 29 days while the chatbot was in operation. Data gathered during this period, including user and submission statistics, are included in this paper to demonstrate the effectiveness of the chatbot and show that it is a suitable alternative for acquiring training data for low-resource languages.

Keywords: *Natural language processing, multi-word expressions, FSTs, UDAR, gamified crowdsourcing*

CHAPTER 1: INTRODUCTION

In an interview with Adriana Belletti and Luigi Rizzi, Noam Chomsky, the father of modern linguistics, said the following: “If you want to make sure that we never misunderstand one another, for that purpose language is not well designed, because you have such properties as *ambiguity* (Chomsky et al., 2002). His words could not be truer in the case of multi-word expressions (MWEs).

MWEs have been defined as “linguistic objects formed by two or more words that behave like a ‘unit’ by displaying formal and/or functional idiosyncratic properties with respect to free word combinations” (Masini, 2019). They can either be compositional or non-compositional/phrasemic (e.g., *break a leg!* is usually expressed as an idiom that means *good luck!*). Natural language processing (NLP) applications, varying from email filters to language translation programs, have some flaws in how they process and differentiate MWEs according to their meanings. These flaws are more prevalent in programs whose models are less mature (i.e., have not been trained on a lot of data) because language models lack a deep semantic understanding of the language they are working on (Eryiğit, Şentaş, & Monti, 2022). To create neural nets that tackle the problems caused by their inherent inadequacy, models require vast quantities of high-quality and diverse training data. However, finding this training dataset is easier said than done. Although there are many sentences on the internet, including MWEs that language models could train on, issues of data quality and diversity arise. Associate Professor Gülşen Eryiğit and masters’ candidate Ali Şentaş proposed a method that allows for the cultivation of sizable amounts of reliable data from differing sources by implementing gamified crowdsourcing into a Telegram chatbot. A brief demonstration of the Russian version of this bot and how it incorporates the principle of gamified crowdsourcing into its usage is provided later in this research paper.

The chatbots developed by Eryiğit and Şentaş accept submissions in the Turkish, English, and Italian languages. Similar to Turkish, which accommodates a complicated morphology, Russian is known to be another morphologically rich language (MRL), and this morphological complexity creates many obstacles for foreigners trying to learn the language. There are three genders, two plurals, and six cases in Russian morphology, with the anticipated exceptions and phonological variations (Leaver, Rifkin, & Shekhtman, 2004). Creating a chatbot that accepts Russian submissions would create many challenges for developers, who need to understand, in simple terms, how Russian morphology works. The developers also need to find suitable toolkits that work with the Russian language (of which there are few).

This research article documents my journey in researching the technical necessities required for the development of the parsing/lemmatizing programs that act as the foundations of the chatbot, operation and maintenance of the chatbot during the 30 days it was active, and retrieval of the statistical data built up during this period. The following paragraphs discuss the structure of this research paper.

In the second chapter, the UDAR finite-state transducer for Russian that was discovered as part of my extensive research is described. Its performance to that of other famous Russian finite-state transducers. While this finite-state transducer is not directly related to the development process of the chatbot, it is integral to the functioning of NLP toolkits.

In the third chapter, the Stanza toolkit, which is used to develop the Russian version of the chatbot, is discussed.

In the fourth chapter, the development process of the Russian version of the chatbot is described. The integration of the Stanza toolkit into the parsing/lemmatization programs is demonstrated.

In the fifth chapter, the essential statistics that were generated during the period the chatbot was active are presented. The trends identified in the statistics are described.

Reference Table for Russian Transliteration

Since much of the focus of this paper is on Russian idiom corpora construction, it is beneficial to include a Cyrillic-ISO 9 transliteration table for easy reference.

Table 1: Cyrillic to ISO 9 transliteration table adapted from bioone.org

Cyrillic	ISO 9:1995	Cyrillic	ISO 9:1995	Cyrillic	ISO 9:1995
А а	A a	К к	K k	Х х	H h
Б б	B b	Л л	L l	Ц ц	C c
В в	V v	М м	M m	Ч ч	Č č
Г г	G g	Н н	N n	Ш ш	Š š
Д д	D d	О о	O o	Щ щ	Ŝ ŝ
Е е	E e	П п	P p	Ъ ъ	” ”
Ё ё	Ë ë	Р р	R r	Ы ы	Y y
Ж ж	Ž ž	С с	S s	Ь ь	’ ’
З з	Z z	Т т	T t	Э э	È è
И и	I i	У у	U u	Ю ю	Û û
Й й	J j	Ф ф	F f	Я я	Â â

CHAPTER 2: UDAR – A COMPARISON BETWEEN FSTS OF DIFFERENT LANGUAGES

Before proceeding with a discussion on the UDAR finite-state analyzer and how it compares to other finite-state transducers, I provide a brief overview of what a finite-state transducer is and explain its mechanism in simple terms.

2.1: What Are Finite-State Transducers?

Finite-state transducers (FSTs) are very similar to finite state automata (FSAs) with one fundamental difference: FSAs have only one memory tape, while FSTs have two: **input** and **output** tapes. FSAs only read input, while FSTs both read the input language and produce an output in a different language. The uses of these finite-state machines, therefore, differ vastly: FSAs are usually used for recognizing patterns, and FSTs are used for translating between different strings.

In providing official definitions of specific FST models, specific notations or **tuples** are used. These tuples can be labeled in different ways, but in this research paper, the tuples are notated in the following manner:

K – the finite set of states
Σ – alphabet of the input
Γ – alphabet of the output
s (s ∈ K) – set of starting states (in diagrams, the starting states are denoted by a circle with an arrow pointing at it)
F (F ⊆ K) – set of final states (in diagrams, the final states are denoted by double circles)
Δ – the transition function from **K×(Σ∪∅)** to **K×(Γ∪∅)**
Figure 1 shows a simple FST model, and its definition is provided below.

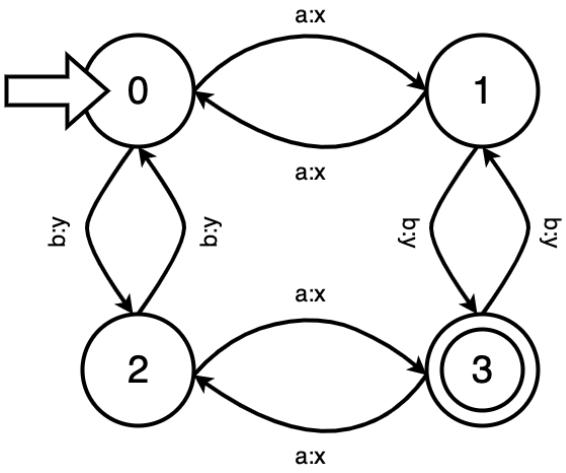

Figure 1: A diagram of a simple finite state transducer

In this FST model,
K = {0,1,2,3},
Σ = {a,b}, *and* **Σ∪∅** = {a,b, ∅},
K×(Σ∪∅) = {(0,a)(0,b)(1,a)(1,b)(2,a)(2,b)(3,a)(3,b),
Γ = {x,y}, *and* **Γ∪∅** = {x,y, ∅},

K×(Γ∪∅) = {(0,x)(0,y)(1,x)(1,y)(2,x)(2,y)(3,x)(3,y)}, and
Δ =

(0,a)→(1,x)	(0,b)→(2,y)
(1,a)→(0,x)	(1,b)→(3,y)
(2,a)→(3,x)	(2,b)→(0,y)
(3,a)→(2,x)	(3,a)→(2,x)

An interesting aspect of finite sets (which FSTs are built upon) is their unexpected utility in real-life applications. I would probably never expect something as trivial as this concept to basically lay the foundation for the small mechanisms that together operate major apps like Google Translate and Siri.

2.2: UDAR: The All-Inclusive Finite-State Analyzer of Russian

Now that our brief overview of FSTs is complete, let us discuss something more angular that is more closely related to our research.

I stumbled upon UDAR during my research on Russian NLP. Realizing that finite-state analyzers like this help create powerful Python NLP toolkits like Stanza, which I review in the upcoming chapter, I decided to scrutinize the UDAR finite-state morphological analyzer and understand how it operates. UDAR, according to its inventor Robert Reynolds, is a finite-state morphological analyzer of Russian that handles stressed wordforms. It utilizes two two-level formalisms: the `lexc` and the `twolc` languages. These languages are used for "creating the lexical network of underlying forms and for realizing orthographic and morphophonological rules on the underlying forms to produce well-formed surface forms," respectively (Reynolds, 2016).

Figure 2, taken from the GitHub page of UDAR, provides a simple example of how UDAR lemmatizes and shows the extra lexical features that turn an individual word into the format it is expressed in in the input sentence.

```
import udar
doc1 = udar.Document('Мы удивились простоте системы.')
print(doc1)
# Мы    мы+Pron+Pers+Pl1+Nom    0.000000
#
# удивились     удивиться+V+Perf+IV+Pst+MFN+Pl  5.078125
#
# простоте      простота+N+Fem+Inan+Sg+Dat      4.210938
# простоте      простота+N+Fem+Inan+Sg+Loc      4.210938
#
# системы       система+N+Fem+Inan+Pl+Acc       5.429688
# системы       система+N+Fem+Inan+Pl+Nom       5.429688
# системы       система+N+Fem+Inan+Sg+Gen       5.429688
#
# .     .+CLB   0.000000
```

Figure 2: An example of how UDAR's Python library processes a Russian sentence

For example, for the word системы (/sɨ'sʲtʲemɨ/), UDAR produces three lexical analyses. For the first analysis, the program initially states the lemma of the word, and then states its lexical characteristics, like it being a noun, its gender, its inanimation, its plurality, and its accusative case. The other analyses, however, claim that the word системы can be composed of different characteristics: the second analysis claims that the word is expressed in the nominal

case, while the third analysis makes an even more contrasting objection by stating that the word is a singular noun and is expressed in the genitive case.

2.3: Comparison of UDAR with Other Russian Morphological Analyzers

The last two decades have seen the development of several Russian morphological analyzers. Many of these analyzers are useful in their respective ways; however, they still lack some attributes that would otherwise make them powerful tools. UDAR, on the other hand, encompasses the most compelling components of these tools and serves them in one package. The table below illustrates the advantages that UDAR has over other analyzers.

Table 2: An example of how UDAR's Python library processes a Russian sentence.

	Year	Platform/OS	Stress	FOSS	Gen.	Disamb.
RUSTWOL	1997	Unknown	-	-	+	-
StarLing	2003	DOS/Windows	+	-	+	-
DiaLing/AOT	2003	Windows/Linux	+	+	+	-
Pymorphy2	2008+	Python (any OS)	-	+	+	-
Mystem3	2003+	All major OSes	-	-	+	+
mocky	2008	Linux/Win/Mac	-	+	-	+
UDAR	2015+	Linux/Win/Mac	+	+	+	+

In Table 2, the plus sign after the year of release for the morphological analyzer indicates that it is under active development. The platform/OS column indicates the platform and/or operating system on which the morphological analyzer functions. The FOSS column clarifies whether the analyzer is free and open source. The stress column indicates whether the system can intelligently analyze or generate stressed wordforms. The Gen. column indicates the analyzer's ability to generate wordforms, and the Disamb. column classifies the analyzer based on its ability to disambiguate multiple readings of a token based on sentential context.

CHAPTER 3: THE STANZA TOOLKIT & THE LEMMATIZATION PROCESS

Finite-state transducers and analyzers like UDAR are useful in determining and documenting the lexical characteristics of words. Analyzers of the Russian language are especially impressive since they undertake the arduous job of categorizing words from morphologically complex Russian sentences into certain attributes. These tools, or at least the theory behind them, lay the foundation for extremely powerful NLP toolkits like Stanza, which is used in creating the submission validation system for the chatbot. In this chapter, I briefly discuss the Stanza toolkit and then provide an overview of the lemmatization process of the submissions.

3.1: What is Stanza?

Stanza is a "collection of accurate and efficient tools for the linguistic analysis of many human languages" (Qi et al., 2020). It is basically a toolkit that can identify sentences and words, their lemmas (base forms), and parts of speech and morphological features. Stanza is available in Python as an importable library and supports more than 70 languages. We initialized the Russian neural pipeline to process the submissions and lemmatize them.

3.2: Technical Background of the Lemmatization Process

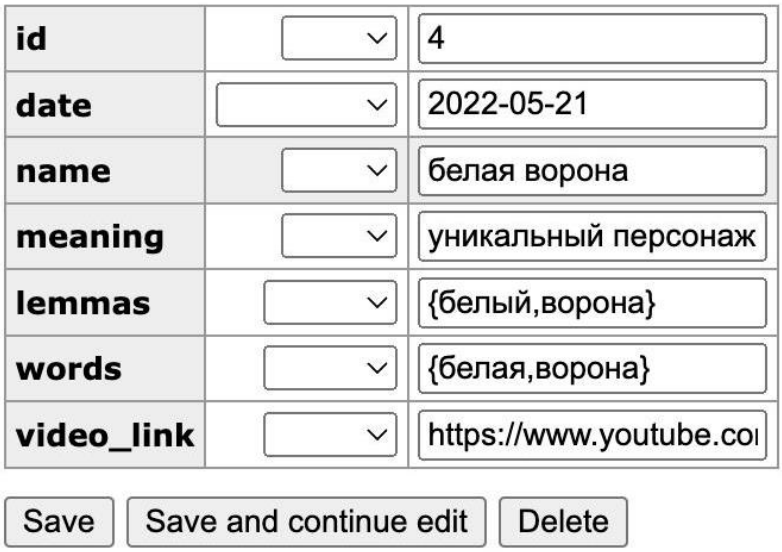

Figure 3: A sample MWE along with its ID number in the database, the date it was assigned to, the meaning of the idiom in Russian, the list of lemmas and the list of words in the MWE, and a video link that explains the idiomatic meaning of the MWE.

All the MWEs, submission, review, and user data were stored in a secure PostgreSQL database. Through this database, we were able to assign an MWE to a certain date so that the bot could send that MWE to the players on that date. Along with the MWE, some complimentary data (which were also sent to users) were stored: the idiomatic meaning of the MWE and a

YouTube video explaining it. However, there was one piece of data that was not shared with the users – the list of the lemmas of the MWE, as shown in Figure 3.

A verb in the English language, like *see*, should cater to many tenses and be changed accordingly: seeing, saw, seen, etc. However, the lemma (base form) of the verb always remains the same. This makes the lemma a great reference for checking whether the words from the MWE of the day are in a submission.

```
import stanza

stanza.download("ru")
nlp = stanza.Pipeline("ru")

doc = nlp("Чтобы выкрасить забор в лиловый цвет, нужно быть полностью без царя в голове.")
#sample input
lemma_list = []
for sentence in doc.sentences:
    for word in sentence.words:
        lemma_list.append(word.lemma)
print(lemma_list)
```

Figure 4: Small program that demonstrates how Stanza is used to lemmatize input samples.

The logic from here is easy to deduce. First, the individual words of the MWE are lemmatized manually and later included in the metadata. As soon as a submission is sent in, a chunk of code like the one in Figure 4 is run, lemmatizing every word in the submission and adding these lemmas to a Python list. Later, an iterative loop checks whether the lemmas of the MWE are on the list of the lemmas of the submission. If yes, the user is prompted to respond whether the MWE is expressed idiomatically or literally. If no, the user is asked to send the submission again.

CHAPTER 4: DODIOM RU CHATBOT DESIGN & GAMEPLAY

4.1: Design of the Chatbot

The Dodiom chatbot was localized in the Turkish, English, and Italian languages in the first release of the research on the chatbot. However, a separate research paper on Russian localization was deemed necessary, as the Russian language uses notably different Cyrillic script and possesses a more complex morphology.

In essence, the game prompts players to write sentences, including an MWE sent to the player every day the game operates. The player, in turn, submits sentences including the MWE of the day and classifies whether it is expressed in a literal or idiomatic meaning. Players can review other players' submissions and leave positive or negative ratings. As a player reviews more samples, they gain more points, and as a submission author gains more positive reviews, they also gain more points. Usually, the reviewer gets one point for every review they give, and an author gets ten points for every positive review they receive on one of their submissions. The user with the most points at the end of the chatbot's operating hours (11 a.m. to 11 p.m., all days of the week) is deemed champion of the day and receives a reward with monetary value. Players who pass thresholds, such as sending a certain number of submissions in a day, win achievements. The chatbot was launched in Azerbaijan, and the daily reward was a 15₼ gift card from the Wolt food delivery app.

The explanation of the game above matches the definition of crowdsourcing – employing a vast group of dispersed people to contribute to a project in exchange for payment. It also fits the structure of a competitive game; hence, it is called a gamified chatbot.

It is important to note that every new player was notified of the fact that the bot was developed for research purposes, and that the samples they submitted could be used for training models. They were also informed that no personal data were processed or made public.

4.2: Gameplay

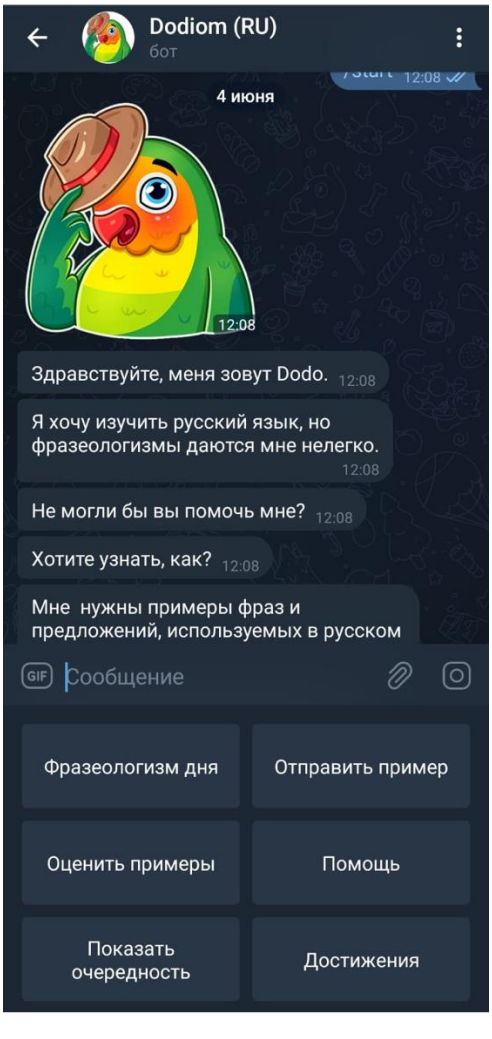

Figure 5: The bot introduces himself and tells the player his purpose.

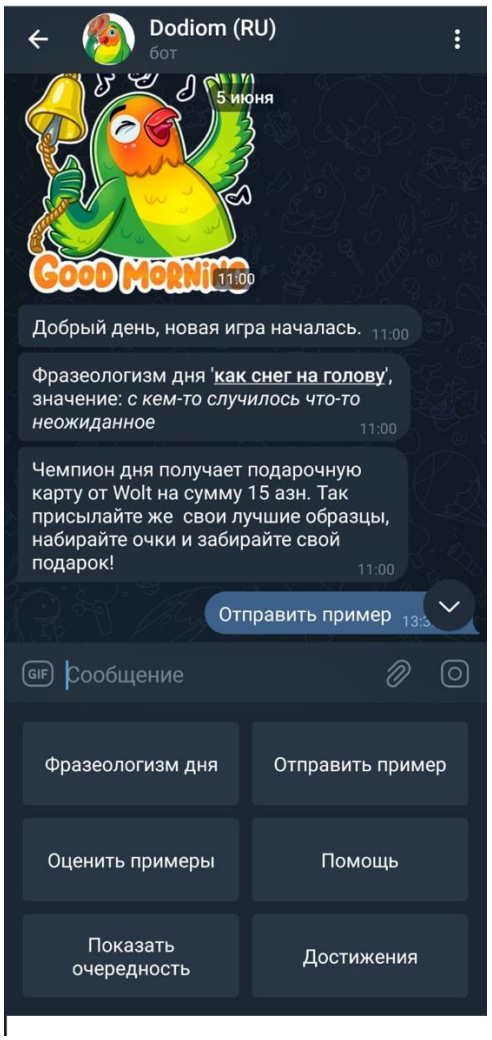

Figure 6: The bot sends the MWE of the day to the player.

Figures 5 and 6 shows what a first-time player would see as they initialize the bot by typing in the `/start` command and how the bot announces its daily MWE at 11 a.m. on each day of operation, respectively. In Figure 5, Dodo, the mascot of the game and the "main character" that the player must message to submit samples, introduces himself. He says that he has difficulties with idioms in Russian and therefore needs the player's help. He then explains in figure 6 how the player can submit samples using the commands available on the command menu, showing a tutorial of the game. The menu is visible in both figures. The translations of the commands on the menu, clockwise starting from the top left command, are: "Phraseme of the day," "Submit an example," "Help," "Achievements," "Show leaderboard," and "Review examples."

As mentioned in Section 4.1, the game sends a new MWE every day the game operates, as shown in Figure 6. If a player cannot view the MWE of a particular day, they can run the "Фразеологизм дня" (/frəzʲɪɐlɐˈgʲizm ˈdnʲa/ – idiom of the day) command by clicking or pressing on it. This command will return the MWE of the day along with an explanation and, in some instances, an explanatory video.

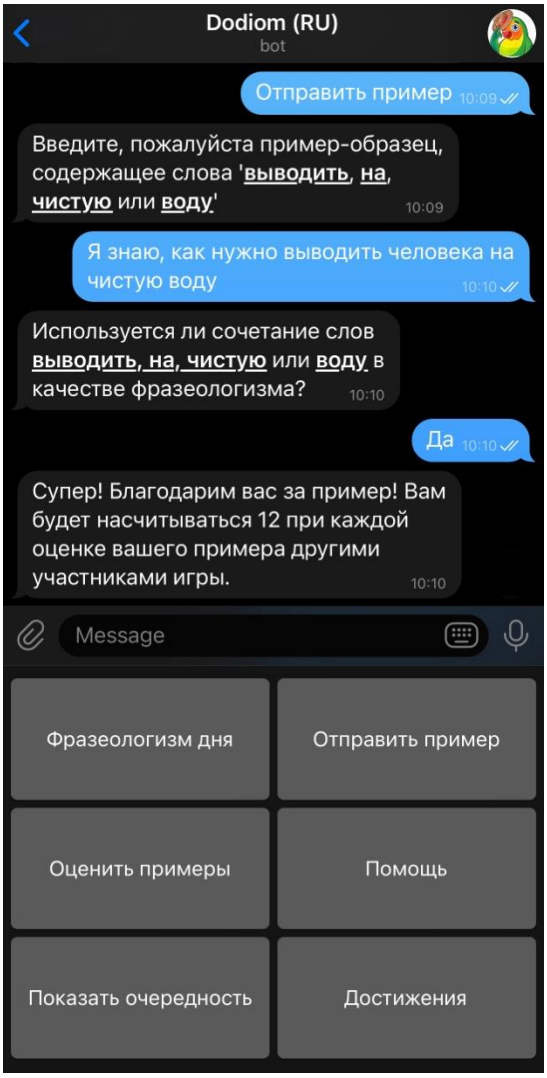

Figure 7: Screenshot showing the process of submitting a sample.

Activating the "Отправить пример" (/ɐtˈpravʲɪtʲ ˈprimɛr/ – submit an example) command enables the player to submit a sample. When they press/click on the command, Dodo first prompts the player to enter a sentence that contains the words in the MWE of the day. Immediately afterward, using Stanza's lemmatization tool, the chatbot checks if the lemmas of the words in the MWE are in the sample (by lemmatizing every word in the sample and comparing). If no issue is found in this step, Dodo asks whether the MWE in the entered sample is expressed in a literal or phrasemic format. The sample is then uploaded to the database for the other players to review. Each time a submission receives a positive review, the author is awarded points. In this way, players are encouraged to submit high-quality samples to obtain more points and get closer to winning the daily reward.

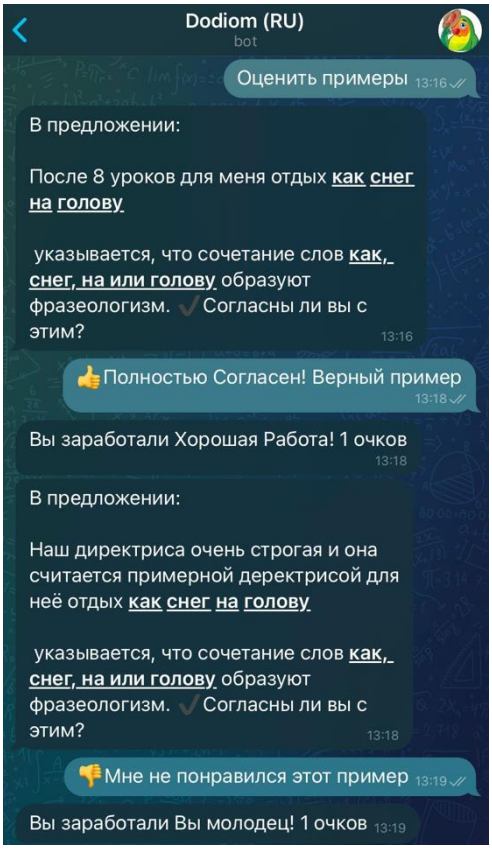

Figure 8: A user expresses a positive review for one submission and a negative review for another.

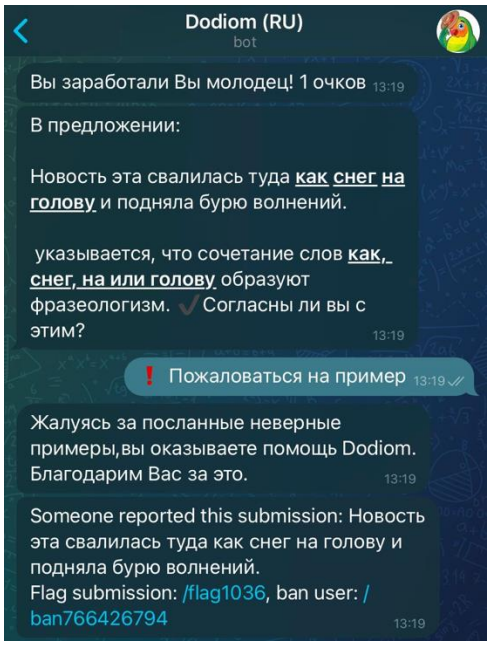

Figure 9: A random sample is reported. The bot shows appreciation for the report, and then sends the developer a message, giving them the option to ban the submission author.

The “Оценить примеры” (/ʊ͡tsɨˈnʲitʲ prʲɪˈmʲerɨ/ – review examples) command allows a player to review other players’ submissions. When this command is selected, the bot sends one submission at a time along with a note stating whether the MWE in the sample is expressed in a literal or non-compositional form. The user is then presented with four choices: leave a positive review, leave a negative review, report the submission, or stop reviewing. If the reviewer chooses one of the first two options, they are awarded a point for their contribution, and Dodo displays another submission for the player to review (if available). It is important to note that submissions with the least reviews are shown to players first to equalize the number of reviews on all submissions. If the reviewer finds a submission inappropriate (e.g., undesirable language, unsuitable use of the bot, same sample submitted multiple times), they can choose the third option and report the submission to the developers. The developers can then flag reported submissions and ban the submission’s author based on the severity of the issue. Note that Figure 9 was taken from the Telegram app of a developer (me), which is why the chatbot gives the user an option to ban the reported submission’s author.

Figure 10: The bot informs the user that for a short amount of time, submitting samples rewards 15 points instead of 10.

Some exceptions are made to the scoring system during *golden minutes*. In these short periods of time, the number of points gained from every positive review on a submission increases from 10 points to 15 points, inspiring players to interact more.

The “Помощь” (/ˈpomɘɛ:/ – help) command sends the user a shortened version of the pre-game tutorial.

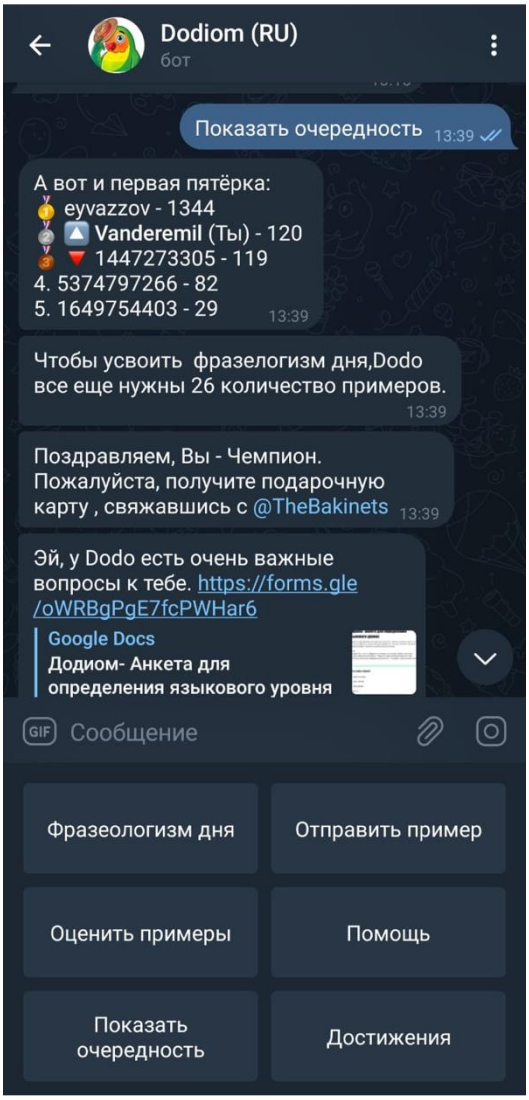

Figure 11: The chatbot sends a list of the first five people in the leaderboard in response to the "Show Leaderboard" command.

Figure 12: The chatbot notifies the player instantly if they are bumped out of the first five list on the leaderboard.

Figure 13:The chatbot informs the user if they achieve the first rank in the scoreboard.

The "Показать очередность" (/pəkɐˈzatʲ ʊ͡tɕɪˈrʲədnəsʲtʲ/ – show leaderboard) command tells the bot to send the list of the top five players with the highest scores at the time the command is sent. In addition to sending this list, the bot sends the number of examples to be sent until the daily soft target of 100 submissions is reached (in Figure 11, this number is 26). The chatbot also notifies players instantly when they are bumped out of the list of the top five players, when they are in first place on the leaderboard, and when they are replaced as the leader by another player.

CHAPTER 5: ANALYZING OBTAINED DATA

After the operation of the game was halted, all relevant data were extracted and visualized using Python's Matplotlib library. In these visualizations, data from Dodiom RU were compared with data from the English version of the chatbot (operated in February of the same year). It is important to note that while champions of all daily games in Dodiom RU were rewarded with items of monetary value, the operative duration of Dodiom EN was divided into two phases, and no rewards of monetary value were provided during the first phase. In this context, the first phase of Dodiom EN is labeled 'EN', while the second phase is labeled 'ENw/Mon.Reward.'

5.1: User Statistics

Dodiom RU's operations spanned from 20 May 2022 to 18 June 2022. Dodiom EN's operation spanned from 14 February 2022 to 8 March 2022, with the first phase ending on 23 February and the second phase starting the day after.

Table 3: Daily Play Usage count for the Dodiom RU and Dodiom EN chatbots based on the number of days from the first day of operation

Number of days	Russian version of the Dodiom Chatbot	English version of the Dodiom Chatbot
1	26	47
2–3	19	19
4–7	6	6
>7	8	4

Table 3 depicts the daily play usage of the two respective chatbots. The numbers indicate the unique number of people who played the game (by either submitting samples, reviewing submissions, or both) over a certain number of days from the first day of operation. For example, while 26 people interacted with Dodiom RU on the first day, only 8 people interacted with the game from day 7 onwards.

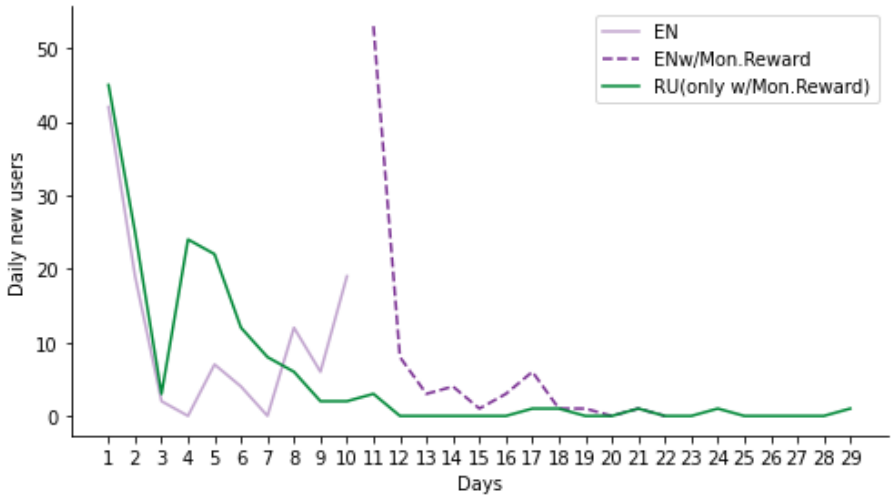

Figure 14: The graph of daily new users registered to respective chatbots across 29 days.

Figure 14 shows the number of new users who started playing the game on a given day over a period of 29 days. The graph of "ENw/Mon.Reward" has its first point plotted on day 11. This was done to more clearly demonstrate the differences between the two different phases of the bot. Expect this type of plotting in the following visualizations.

The "ENw/Mon.Reward" and the "RU (only w/Mon.Reward)" graphs follow an expected pattern: a very high number of new users on the first day, followed by decreasing numbers of new users in the ensuing days. This pattern closely resembles an exponential function whose coefficient is between 0 and 1. The only exception is the graph labeled 'EN,' where, toward the end of the phase, the daily new user count starts increasing.

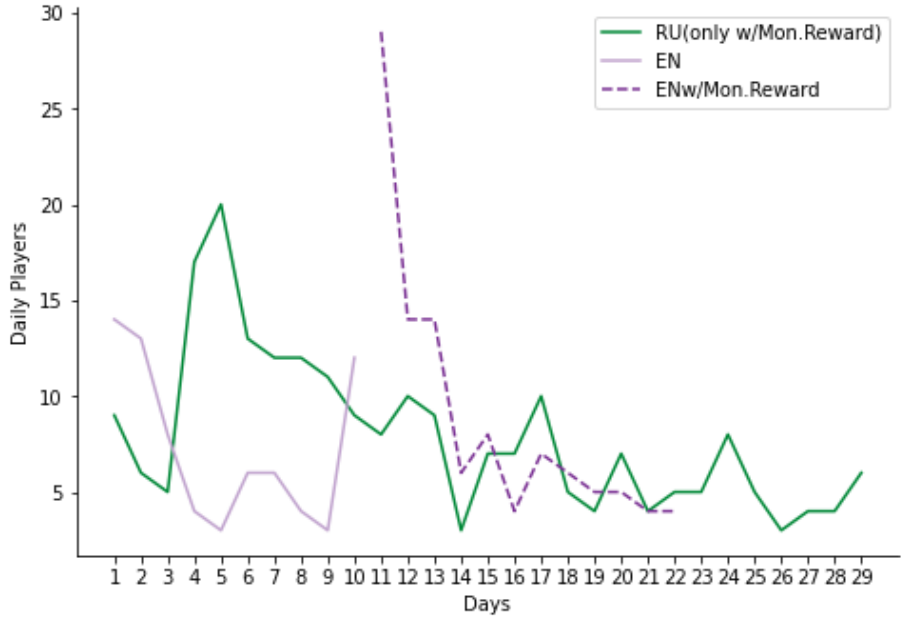

Figure 15: Daily number of players who interacted with respective chatbots over 29 days.

Figure 15 shows the number of daily users who played the game over 29 days. Compared to Figure 14, the number of daily users does not plummet to extremely low values after a few days of operation; it seems to oscillate, although decreasing in magnitude as days pass.

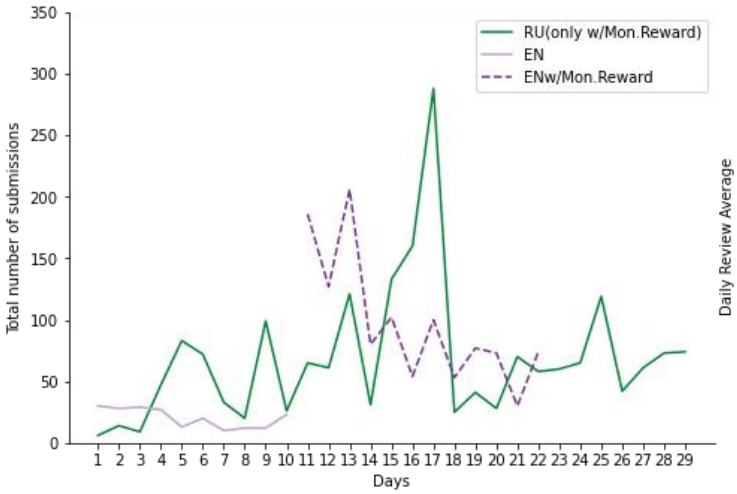

Figure 16: Total number of submissions per day.

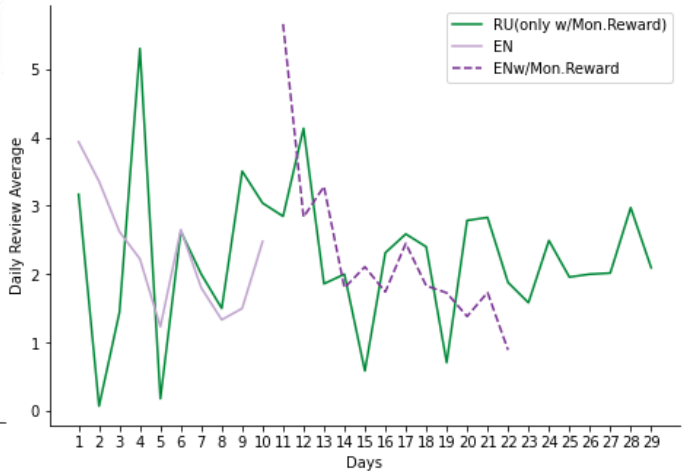

Figure 17: Daily review average.

Figure 16 shows the total number of submissions sent by all users per day over a 29-day period, while Figure 17 shows the daily review average (total number of reviews made by users that day divided by the total number of users) per day across the same period. In both figures, the graph for Dodiom RU exhibits an especially peculiar behavior. While the graphs for both phases of Dodiom EN display a gradual descent in value, the Dodiom RU graph continues to oscillate without decreasing to an absolute minimum at the end of the period. We believe that the cause of this was the continual advertisement of the monetary rewards of the game, both in-game and on other platforms.

5.2: Submission Statistics

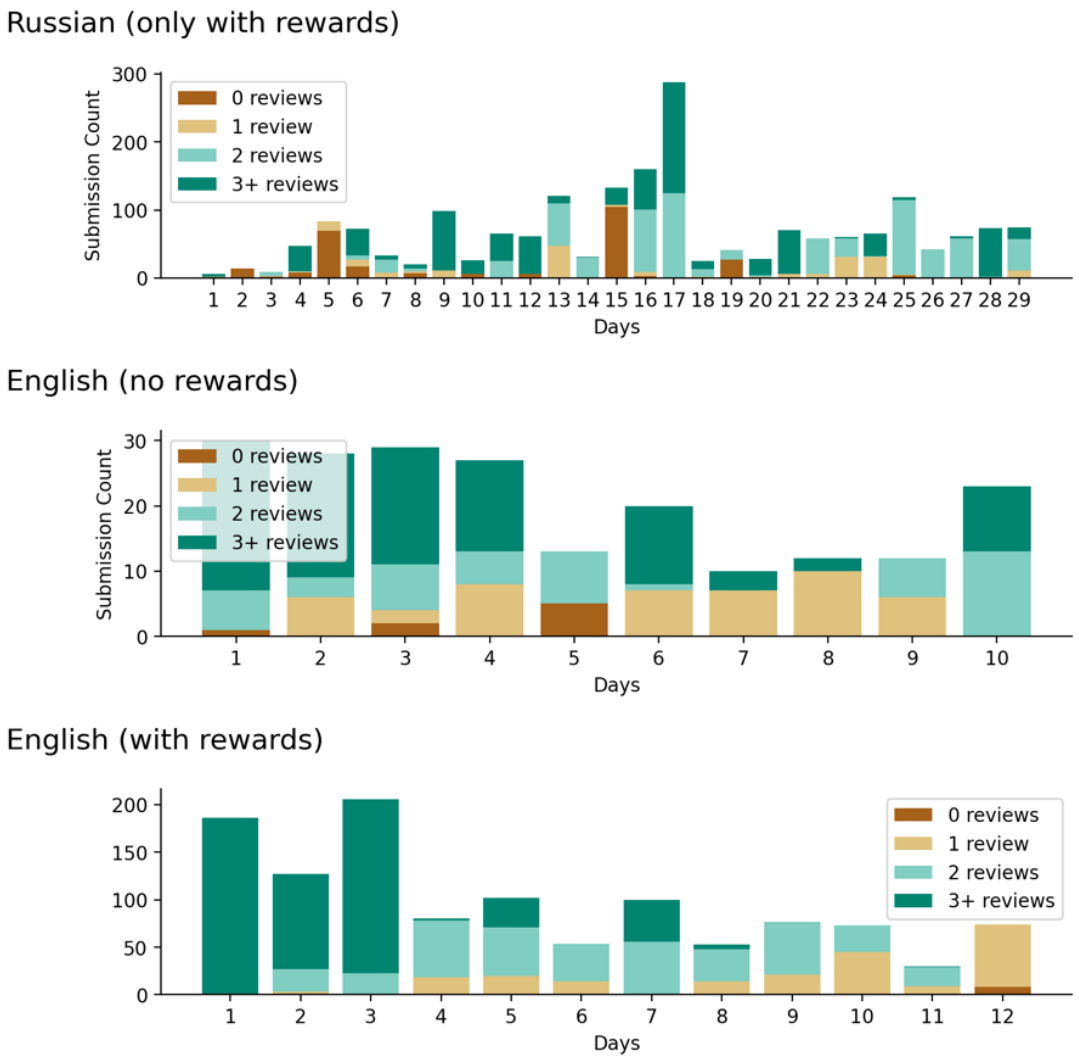

Figure 18: Frequency of reviews on submissions per day.

Figure 18 illustrates the frequency of reviews of submissions on a given day. The y-axis shows the number of submissions. For example, on the 27th day of Dodiom RU's operation, a sizable majority of all submissions (around 70) received two reviews, with the rest receiving three or more reviews. On the following day, all submissions (around 80) received at least three reviews.

An interestingly counterintuitive pattern is exhibited by the plot for Dodiom RU. While the number of submissions and reviews decreases with every passing day for the second phase of the Dodiom EN bot, for Dodiom RU, these numbers seem to increase. For the first few days, the numbers of both submissions and reviews are at all-time lows, but with each passing day, these numbers grow larger. I believe the explanation for this behavior is that during the later stages of the game, we started to remind players every day through the chatbot that a reward was waiting

for the champion at the end of that day. This could have motivated players to start playing more frequently in the last two weeks of Dodiom RU's operation.

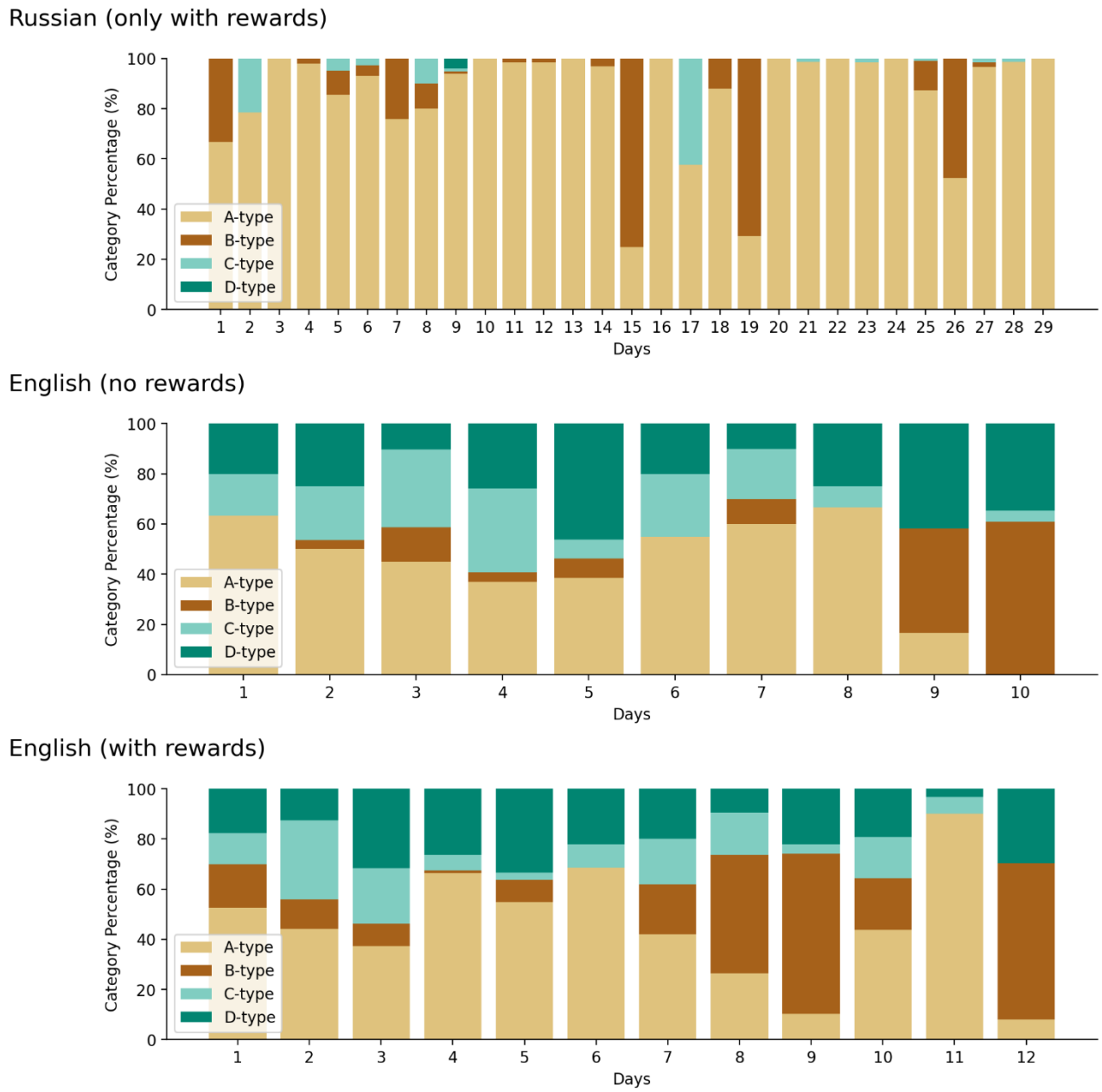

Figure 19: Categories of submissions per day.

Figure 20 shows how daily submissions are divided into four categories based on the structure and syntax of the MWEs present in the submissions. The A-type category represents all the MWEs that have been expressed in an idiomatic manner, with all the words of the MWE together and are not separated by other words in the sentence. An example of an A-type submission in English would be, "For the first time in his life, Raymond *hit the books* and

studied for today's test." A B-type submission contains idiomatic MWEs that are separated by words, such as adjectives, or even phrases. For example, "Adriana *hit the* wrong *books* for today's exam."

The C- and D-type submissions are similar to the A- and B-type submissions, respectively, with one key difference: the C- and D-types consist only of non-idiomatic submissions. An example of a C-type submission would be, "She was so furious that she started *hitting the book* on her desk," and an example of a D-type submission would be, "He *hit* a student with *the* chemistry *book* and thereupon got a lunch detention."

CHAPTER 6: CONCLUSION AND FUTURE WORKS

Languages worldwide foster an attribute of ambiguity, as mentioned in Chapter 1. Ambiguity is the reason language models have a hard time processing MWEs, in which expressions are sometimes non-compositional. This issue can be improved by acquiring better training data for these models. Gamified crowdsourcing can meet this need by utilizing "the wisdom of the crowd" and encouraging competition to obtain high numbers of quality samples. The user and submission/review statistics serve to demonstrate this idea. For their input, the best players were incentivized with gift cards, which proved to be a great source of motivation.

This research paper partly achieves the goal defined by the inventors of the Dodiom chatbot in the conclusion of their research paper: extending the scope of the game to other languages. Hopefully, this game will be available in more languages in upcoming years, contributing to the development of this research field.

The purpose of using gamified crowdsourcing in this research was to obtain sizable quantities of high-quality submissions for training language models. It is important to note, however, that crowdsourcing is not only limited to this aspect of NLP.

For example, my latest start-up project, *Mektebim S2C*, which is taking place under my non-profit organization *Lexically Curious*, has successfully merged one of the most recognized PPP programs in Azerbaijan, "Mektebim" (translation: my school), with the corporate social responsibility (CSR) project of one of Azerbaijan's leading fast-moving consumer goods companies, *Azersun*.

It took considerable time and effort to choose the right theme for the project and the right Azersun product on which to establish it. However, NLP can be implemented in the procedure of determining themes for CSR, making it automated, less time-consuming, more effortless, and economically viable. In India, conglomerates are required by law to allocate a portion of their budgets to CSR projects if their revenue passes a certain threshold. To effectively pick a theme for a CSR project, these conglomerates employ advanced crowdsourcing methods and text-mining models on the annual reports of several Indian companies (Parimi et al., 2020). I strongly believe that employing NLP techniques and crowdsourcing approaches could define the future of determining the most effective CSR themes in the business realm.

ACKNOWLEDGMENTS

I would primarily like to express my sincere gratitude to both Associate Professor Gülşen Eryiğit and Ali Şentaş from Istanbul Technical University for letting me expand the range of their initial research and for providing ongoing support in my endeavors. I would also like to thank Bahadir Yeltekin (my father) for financially supporting this research project and helping with the management of the chatbot and the structure of this research paper, Elnara Yeltekin (my mother) for translating the messages Dodiom sends from Turkish and English to Russian and for the creation of the Russian MWE list, and Emil Muxtarov and Aysel Seyidova for helping with the graphical design and further development of the Russian MWE list. Finally, I would like to thank the users of the Dodiom RU chatbot for continually providing the submissions and reviews that made this project happen.

REFERENCES

Chomsky, N., Rizzi, L., & Belletti, A. (2002). An interview on minimalism. In N. Chomsky, *On Nature and Language* (pp. 92-161). Cambridge: Cambridge University Press. https://doi.org/10.1017/CBO9780511613876

Eryiğit, G., Şentaş, A., & Monti, J. (2022). Gamified crowdsourcing for idiom corpora construction. *Natural Language Engineering*, 1–33.

Leaver, B. L., Rifkin, B., & Shekhtman, B. (2004). Apples and oranges are both fruit, but they do not taste the same: A response to Wynne Wong and Bill VanPatten. *Foreign Language Annals*, 125–132.

Masini, F. (2019, September 30). Multi-word expressions and morphology. In *Oxford research encyclopedias*. https://oxfordre.com/linguistics/view/10.1093/acrefore/9780199384655.001.0001/acrefore-9780199384655-e-611?print=pdf

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., & Manning, C. D. (2020). Stanza: A Python natural language processing toolkit for many human languages.

Reynolds, R. J. (2016). Russian natural language processing for computer-assisted language learning. Tromsø: UiT The Arctic University of Norway.

R, N., Parimi, M. R., S, A. R., NS, N. K., & Babu, S. (2020). Develop CSR themes using text-mining and topic modelling techniques. In *I. Staff, 2020 IEEE International Conference on Cloud Computing in Emerging Markets* (CCEM) (pp. 67-71). Bengaluru: IEEE.