University of Ljubljana
Faculty of Computer and Information Science
Faculty of Mathematics and Physics

Frenk Dragar

SloBench: Slovenian Natural Language Processing Benchmark

bachelor thesis

Interdisciplinary University Study Programme
Undergraduate programmes
Computer Science and Mathematics

Mentor: Assist. Prof. dr. Slavko Žitnik

Ljubljana, 2022

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Fakulteta za matematiko in fiziko

Frenk Dragar

SloBench: Slovenski vrednotnik metod za obdelavo naravnega jezika

diplomsko delo

Interdisciplinarni univerzitetni
študijski program prve stopnje
Računalništvo in matematika

Mentor: doc. dr. Slavko Žitnik

Ljubljana, 2022

The text is formatted with LaTeX.

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Candidate: Frenk Dragar
Title: SloBench: Slovenian Natural Language Processing Benchmark
Thesis type: Bachelor Thesis
Mentor: Assist. Prof. dr. Slavko Žitnik

Description:
With the recent rise in popularity of transformer-based natural language processing models and their state-of-the-art performance on many NLP tasks, there is an increasing need to objectively evaluate these tools and enable their comparison. There exist a number of datasets and benchmarks for NLP tasks, but they are mostly based on the English language. In this thesis, the candidate describes the creation of the first Slovenian automatic NLP benchmarking platform - SloBench, along with its underlying model-agnostic extensible evaluation framework. The project is then critically evaluated and compared to existing NLP benchmarks. Finally, some ideas for future extensions to the platform are provided.

Kandidat: Frenk Dragar

Naslov: SloBench: Slovenski vrednotnik metod za obdelavo naravnega jezika

Vrsta naloge: Diplomska naloga na interdisciplinarnem univerzitetnem programu prve stopnje Računalništvo in matematika

Mentor: doc. dr. Slavko Žitnik

Opis:

Z nedavno priljubljenostjo modelov obdelave naravnega jezika, ki temeljijo na arhitekturi transformer in njihove najsodobnejše zmogljivosti pri številnih nalogah NLP, je vse večja potreba po objektivnem ocenjevanju teh orodij in omogočanju njihove primerjave. Obstajajo številni nabori podatkov in meril za NLP naloge, ki pa večinoma temeljijo na angleškem jeziku. V diplomski nalogi kandidat opiše razvoj prve slovenske platforme za avtomatsko primerjavo NLP modelov - SloBench, skupaj z njenim razširljivim in od sistemske arhitekture neodvisnim ogrodjem za evalvacijo sistemov. Nato kritično oceni projekt, ga primerja z obstoječimi merili uspešnosti NLP in poda nekaj idej za prihodnje razširitve platforme.

I would like to thank doc. dr. Slavko Žitnik for his mentorship during the planning and development of the SloBench project, along with the writing of this bachelor thesis.

I would also like to thank my parents, sister, and Kristi for their support during my studies.

Thanks also goes to the members of the community of Slovenian Developers, who helped me discover and make sense of the world of Django development.

Contents

List of abbreviations

Abbreviation	Meaning
NLP	natural language processing
NLU	natural language understanding
NLG	natural language generation
MT	machine translation
NER	named entity recognition
QA	question answering
BLEU	bilingual evaluation understudy
ROUGE	recall-oriented understudy for gisting evaluation
METEOR	metric for evaluation of translation with explicit ordering
GLUE	general language understanding evaluation
KLEJ	kompleksowa lista ewaluacji jezykowych
SQuAD	Stanford question answering dataset
MOROCCO	model resource consumption
XTREME	cross-lingual transfer evaluation of multilingual encoders
DB	database
JSON	javascript object notation
TSEO	task submission evaluation object

Abstract

Title: SloBench: Slovenian Natural Language Processing Benchmark

Author: Frenk Dragar

Evaluation of natural language processing (NLP) tasks is an essential part of research and progress in the field. It provides an objective standard for comparison and performance on a specific task. We give an overview of recent public benchmarks and evaluation trends, with focus on the automatic evaluation of systems. We then propose, implement and document a general and extendable model-agnostic evaluation framework, along with the first online platform for the automatic evaluation of Slovene language NLP tasks with public leaderboards, showing the performance of submitted systems.

Keywords: natural language processing, benchmarking, leaderboard, machine learning, web platform.

Povzetek

Naslov: SloBench: Slovenski vrednotnik metod za obdelavo naravnega jezika

Avtor: Frenk Dragar

Evalvacija nalog procesiranja naravnega jezika (NLP) je bistven del raziskav in napredka na tem področju. Zagotavlja objektiven standard za uspešnost in primerjavo sistemov pri določeni nalogi. Podamo pregled nedavnih javnih lestvic za najboljše sisteme in trendov njihovega ocenjevanja s poudarkom na avtomatskem vrednotenju sistemov. Nato predlagamo, implementiramo in dokumentiramo splošno, razširljivo in od sistemske arhitekture neodvisno ogrodje za evalvacijo sistemov, skupaj s prvo spletno platformo za avtomatsko vrednotenje NLP nalog v slovenščini z javnimi lestvicami, ki prikazujejo rezultate objavljenih sistemov.

Ključne besede: procesiranje naravnega jezika, vrednotenje, lestvica najboljših, strojno učenje, spletna platforma.

Razširjen povzetek

Vrednotenje sistemov za procesiranje naravnega jezika je pomemben del raziskovanja in aplikativnosti tega področja. Brez objektivnih meril uspešnosti modelov ali orodij za določeno nalogo, se ne moramo pogovarjati o napredku, primerjavi in dosežkih rezultatov. NLP je v zadnjih 10 letih doživel ogromno napredka, kar je povezano z razvojem novih metod procesiranja, razpoložljivostjo večjih naborov podatkov in procesorske moči v obliki grafičnih kartic, uporabljenih za učenje globokih nevronskih mrež.

Pri večini NLP nalog (in bolj splošno, strojnem učenju) lahko razpoložljive podatke (recimo korpus) razdelimo na:

- **učne podatke**, najpogosteje v obliki ročno označenih parov vhodnih in izhodnih podatkov, ki se uporabljajo za učenje algoritmov nadzorovanega učenja,
- **validacijske podatke**, označen nabor podatkov, ki se uporabljajo za prilagajanje hiperparametrov modela (recimo število skritih nivojev nevronske mreže),
- **razvojno-testne podatke**, označene podatke ki se lahko uporabijo za samotestiranje sistemove sposobnosti med razvojem,
- **testne podatke**, ki se uporabljajo za končno ocenjevanje sistemove sposobnosti.

Ker želimo, da so rezultati vrednotenja NLP modela neoporečni, se pogosto pri testni množici podatkov odstrani oznake in v javnost deli samo

vhode. To množico podatkov nato NLP model vzame kot vhod v sistem in napravi predvidene *izhode* - ti pa se lahko primerjajo z dejanskimi izhodi testnih podatkov. Nekatere naloge NLP lahko obravnavamo kot klasifikacijski problem. Med te spadajo: označevanje imenskih entitet, sentimentalna analiza, tokenizacija, ... Pogoste metrike za primerjavo takšnih nalog so:

- **natančnost** (angl. *recall*) - razmerje med številom pravilno označenih entitet in vsemi razpoznanimi
- **priklic** (angl. *recall*) - razmerje med številom pravilno označenih in tistimi, ki bi morale biti identificirane
- **metrika F** (angl. *F-measure, F-score*) pa je kombinacija prejšnih dveh metrik v eno samo, kot njuna harmonična sredina

Bolj zapletenih nalog, predvsem tistih, ki so generativne narave, torej kjer je naloga ustvariti besedilo v naravnem jeziku, ne moremo vrednotiti kot problem klasifikacije, torej enostavno primerjati parov vhod-izhod. Dva strukturno in besedno precej različna povzetka istega besedila sta na primer lahko enako dobra pri izluščenju bistva daljšega izvirnega besedila, a ju ne moramo primerjati kar neposredno. Za takšne naloge obstajajo metrike kot BLEU (angl. *Bilingual Evaluation Understudy*), METEOR (angl. *Metric for Evaluation of Translation with Explicit Ordering*) in ROGUE (angl. *Recall-Oriented Understudy for Gisting Evaluation*), ki temeljijo na prekrivanju n-gramov, statistikah najdaljšega skupnega podzaporedja ter več za svoj izračun. Kljub temu da te metrike v večini primerov dobro korelirajo s človeško oceno, je njihova uporaba med raziskovalno javnostjo včasih vprašljiva, saj dvig točk pri teh metrikah ne zagotavlja nujno izboljšave zmogljivosti sistema.

S porastom naprednejših metod za reševanje problemov kot so strojno odgovarjanje na vprašanja, strojno prevajanje, avtomatsko povzemanje daljših besedil, označevanje imenskih entitet, ipd., je potrebno zastaviti sistemom, ki te naloge rešujejo, težje izzive. V zadnjih letih se je zaradi teh potreb

pojavilo precejšnje število platform za merilo uspešnosti oz. vrednotnikov (angl. *benchmark*) kot so GLUE, SuperGLUE, KLEJ, RussianSuperGLUE, XTREME, GENIE, GEM, SQuAD in mnogo drugih. Te se razlikujejo od svojih predhodnikov v tem, da poleg naborov podatkov za ocenjenvanje in relevantnih metrik ponujajo tudi spletne platforme za vrednotenje objavljenih prispevkov. Tako so naloženi rezultati po uspešnem vrednotenju vidni na spletni strani v tabelah, ki se z vsakim novim prispevkom posodobijo in prikazujejo napredek skozi čas. Tako lahko vsakdo spremlja napredek področja, prispeva svoje rezultate in primerja rezultate vseh, ki so jih tam objavili. Tema te naloge je razvoj takšne spletne platforme za vrednotenje sistemov procesiranja naravnega jezika za slovenščino, skupaj z njenim zalednim sistemom za vrednotenje prispevkov.

Evalvacijsko ogrodje, ki ga SloBench platforma uporablja za vrednotenje prispevkov, je zasnovano z razširljivostjo in dolgodobnostjo v mislih. Ne polaga predpostavk in omejitev na postopek vrednotenja prispevka, razen sledečih: vhod v evalvacijsko skripto je ena datoteka (zaenkrat je v vseh primerih to zip datoteka), ki vsebuje izhodne podatke ovrednotenega programa - te je generiral sistem za procesiranje naravnega jezika na podlagi neoznačenih vhodov, ki so na voljo na SloBench platformi (angl. *test dataset*). Izhod pa je poseben JSON objekt - TSEO (angl. *Task Submission Evaluation Object*), ki poleg izračunanih metrik podane naloge vsebuje še nekaj metapodatkov kot so uspešnost evalvacije, morebitno sporočilo napake in čas vrednotenja. Skripte za vrednotenje so zapakirane v Docker zabojnike (angl. *Docker container*), ter lahko vsebujejo poljubne knjižnice in programsko okolje, dočim je njihovo izvajanje možno gostiti v okolju Docker - to v praksi pomeni praktično neomejen nabor okolij za izvajanje programske opreme. V sklopu evalvacijskega ogrodja je implementiranih nekaj skript za branje vhoda, evalvacije, ter pisanja izhodnega JSON objekta za nabor nalog kot so odgovarjanje na vprašanja, skrajševanje besedil, ter strojno prevajanje, z namenom da se lahko te skripte razširijo in prilagodijo bodočim, novim nalogam.

Evalvacijsko ogrodje je osnova za avtomatsko vrednotenje prispevkov na spletni platformi SloBench, ki služi kot spletna platforma za vrednotenje prispevkov nalog v slovenskem jeziku. Spletna storitev je implementirana v okoljih Django za zaledni sistem in React za sprednji del storitve. Platforma ponuja možnost ogleda lestvice najboljših prispevkov (angl. *leaderboard*) za vsako izmed nalog, ki so javno objavljene na storitvi. Za prispevanje lastnih rezultatov se mora uporabnik registrirati in prijaviti v spletno stran, kar mu omogoča prenos nabora podatkov za razvoj lastnih sistemov za opravljanje dane naloge (angl. *train dataset*), primera objave (angl. *sample submission*), s katerim uporabnik vidi v kakšnem formatu mora pripraviti svojo rešitev, ter neoznačenim testnim naborom podatkov (angl. *test dataset*), na katerem lahko požene svoj sistem in pridobi izhodne podatke za objavo na lestvici najboljših.

Spletna platforma SloBench pozna štiri vloge uporabnikov:

- **Opazovalec**, anonimni in neprijavljeni uporabnik, lahko opazuje prispevke na strani oz. lestvicah za razne naloge.
- **Sodelavec**, registriran in prijavljen uporabnik, lahko prispeva svoje rezultate za posamezno nalogo.
- **Urednik** lahko ureja in upravlja oz. moderira svoje naloge in lestvice najvišjih.
- **Administrator** lahko poleg vsega, kar lahko počne urednik, dodaja na stran nove naloge, ter upravlja uporabnike in osnovne lastnosti spletne strani (opis, pogosta vprašanja, ...) preko Django administracijskega vmestnika.

Pri urejanju posamične lestvice lahko urednik ali administrator nastavita osnovne podatke kot so naslov, opis, navodila za objavo, ter nastavitve kot avtomatsko obvestilo o uspešnem oz. neuspešnem vrednotenju avtorju objave po e-pošti in privzeto nastavitev objave na javno ali skrito. Omeji lahko tudi število prispevkov na uporabnika v določeni časovni enoti (dnevu, tednu

ali mesecu), kar preprečuje preveliko prilagajanje sistema na skritih testnih podatkih (angl. *overfitting*). Platforma omogoča tudi tedensko pošiljanje obvestil uporabnikom, ki so se naročili na njih - vsebujejo poročilo on napredkih objav na lestvici najboljših za določeno nalogo v prejšnjem tednu.

Platforma SloBench je torej prvi javni avtomatski vrednotnik sistemov za procesiranje naravnega jezika za slovenščino. Opazovalcem omogoča preprost način pregleda nad napredkom sistemov pri opravljanju določene naloge, razvijalcem takšnih sistemov pa način validacije in objavljanja svojih dosežkov. Trenutno so na spletni platformi objavljene tri lestvice za NLP naloge - odgovarjanje na vprašanja, katerih vir podatkov je prevedena verzija vrednotnika SuperGLUE, ter dve lestvici za avtomatsko prevajanje - iz angleščine v slovenščino in obratno. Pripravljenih je še več evalvacijskih skript, ki pa bodo v bližnji prihodnosti naložene na platformo, ko bodo na voljo končni nabori podatkov za pripadajoče naloge. V nedavnem času so se pojavile tudi nove platforme za vrednotenje, kot so Explainaboard in Dynabench. Te inovirajo na področju NLP vrednotenja z naprednimi metodami primerjeve in analize objavljenih sistemov, ter zbiranju podatkov med samim ocenjevanjem. Te rešitve so zaenkrat še precej nove in nedokazane, podajajo pa ideje za možnosti bodočih razširitev SloBench spletne platforme.

Chapter 1

Introduction

Natural language processing (NLP) has become an integral part of our society, with many people using the technology embedded into our smartphones, voice assistants, search engines, text processing applications, and more, every day. While research in the field moves at a tremendous pace, many of the use cases are far from perfect. How *good* are these applications at performing various tasks? How do systems performing the same task *compare* to each other? How has the performance of this technology *progressed* through time?

Evaluation of natural language processing tasks is an essential part of NLP research. It enables the comparison of various NLP systems, sets measurable goals for their performance in the process of development, checks for biases in outputs, tracks performance over time and gives visibility to the evaluated tools by displaying evaluation results on a public benchmark. By comparing two systems over a set of objective evaluation metrics, one can decide which is preferred for their use case, giving way to better end applications of NLP on real world tasks.

Every NLP task has many valid metrics by which it can be evaluated - some more complex, difficult to perform, requiring more dependant software or human input than others. Regardless of complexity, NLP evaluation can be formulated in a common fashion: the comparison of input-output pairs from *reference* (usually annotated by human experts) and *predicted* (gener-

ated by the evaluated NLP system) data.

The main topic of this thesis is the creation of the first online platform for the benchmarking and evaluation of Slovene natural language processing tasks. For this purpose, we propose a general and extendable system for the evaluation of natural language processing tasks, which is able to work with virtually any software dependencies, input data formats and evaluate any system capable of producing an output file in an arbitrary format, specified by the task creator. It enables users to submit the results of their systems on a particular task's unlabeled test data and publish the score generated by the evaluation job on a public leaderboard, aggregating all published results into a table, sorted by system performance. The online platform then allows observers to compare the submitted systems, analyse their performance based on the evaluated metrics and keep track of performance on a task over time, as new submissions get added.

Chapter 2

Related Work

This chapter first provides a short overview of natural language processing, its development over time and some of its most common tasks. This provides context for the second section, which delves into evaluation of NLP by describing its historical background, terminology and some basic evaluation principles. The third section examines the most popular and influential existing NLP benchmarks in recent history, describing both their innovations and provided online evaluation platforms.

2.1 Natural Language Processing

Natural Language Processing (NLP) is an interdisciplinary field of research and application, joining together computer science, linguistics and artificial intelligence. It explores how computers can be used to understand, generate and manipulate natural language (text or speech) [1]. Its beginnings reach into the late 1940s, starting with simplistic machine translation (MT) using dictionary-lookup and word reordering after translation to fit the target language. This produced poor results, indicating the problem was much harder than it had initially seemed. Noam Chomsky's work on theoretical analysis of language grammars in the 1950s introduced new concepts such as *context free*, *generative*, and *regular* grammars, which provided a theoretical perspec-

tive on language processing using symbolic approaches. Another perspective emerged around the same time, using statistical methods and focusing on speech recognition, but was less popular than its symbolic counterpart.

After the first wave of initial enthusiasm in the field, a report from the ALPAC (Automatic Language Processing Advisory Committee of the National Academy of Science - National Research Council) in 1966 concluded that machine translation was not immediately achievable. This led to cuts in funding of MT and other NLP research in the United States of America and consequently, a decrease in work in the field [2]. Some influential projects around this time include ELIZA, which simulated conversation between between humans using pattern matching and SHRDLU, which simulated a robot that manipulated blocks on a tabletop based on text prompts, using knowledge of a simple virtually constructed environment.

In the early 1980s, when more advanced computational resources were made available, a push for non-symbolic or statistical approaches became more prevalent. There was intent of creating NLP applications that worked in a broader, real-world context. In the following years, the field grew significantly, leveraging more powerful computers and the new abundance of electronic text, especially after the introduction of the internet.

More recently, in the late 2010s, the advent of the semantic web and social media enabled the creation of massive natural language datasets. Enormous parallel computation also became available via graphics processing units (GPUs). In this environment, research became focused on neural networks, producing state-of-the-art results on many NLP tasks with systems like BERT, ELMo and RoBERTa, using novel technologies, such as transformers [3].

2.1.1 Overview of Common Tasks

The common sub-problems in NLP can be roughly categorised into lower-level and higher-level tasks, dealing with lower (phonological, morphological, lexical, syntactic) and higher (semantic, discourse, pragmatic) levels of lan-

guage, respectively. These can be represented visually in the *NLP pyramid* (Figure 2.1).

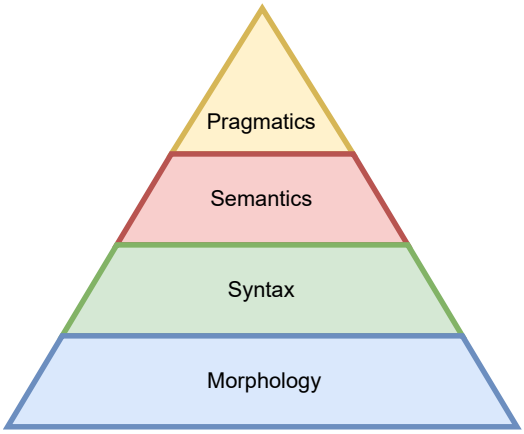

Figure 2.1: NLP pyramid, depicting levels of natural language processing and their relationship, where upper levels build on top of lower ones.

Lower-level tasks include: **Tokenization** - the process of separating a piece of text into smaller units called tokens (words, characters, sub-words), **Part of Speech (POS) Tagging** - the identification of nouns, verbs, adjectives, adverbs, etc., **Morphological decomposition/parsing** - the process of splitting compound words into smaller parts (e.g. roots, prefixes, suffixes, ...), **Shallow parsing (chunking)** - identifying phrases from part-of-speech tagged tokens, **Co-reference resolution** - connecting pronouns and other referring expressions to the right individuals, etc. [1]. Most of these can often be evaluated easily, by defining the task as a classification problem, which is also known in the wider field of machine learning. In such a problem, the evaluated system is tasked with annotating a dataset using various labels or *classes* (e.g. *noun*, *verb*, *adjective*, ...). The most popular metrics for describing classification performance are: *precision* (positive predictive value) - a measure of result relevancy, *recall* (*true positive rate*) - a measure of how many truly relevant results are returned, and *F-measure* (*F-score*) - a measure that combines precision and recall as their harmonic mean.

Higher-level tasks, some of which usually build on top of low-level tasks,

include: **Named entity recognition (NER)** - the identification of specific words/phrases (*entities*) and their categorisation (e.g. people, organisations, locations, time expressions), **Sentiment analysis** - extracting opinion states and subjective information (e.g. from reviews and survey responses, social media, ...), **Text summarization** - summarizing a longer text into a shorter one while keeping most of the meaning, **Question Answering** - generating full natural text replies, possibly using previous or external knowledge, **Automatic/Machine Translation**, etc. [1]. These tasks, often generative by nature, are usually harder to evaluate, as we cannot always simply compare the evaluated system's outputs to a reference dataset. For example - two different summaries of the same text can be equally valid at summarizing the original text, but are not directly comparable, as with e.g. annotation tasks. For this purpose, special metrics and evaluation methods must be created. These include *BLEU* (Bilingual Evaluation Understudy) - an evaluation metric that computes text distances based on tri-grams, *METEOR* (Metric for Evaluation of Translation with Explicit ORdering) - based on the harmonic mean of unigram precision and recall, and *ROGUE* (Recall-Oriented Understudy for Gisting Evaluation) - a method that uses the overlapping of n-grams, longest common subsequence based statistics and more for its metric computation. While these metrics correlate well with human judgment, there has been some controversy surrounding their use, with researchers arguing that there is no guarantee that an increase in score indicates system quality [4].

2.2 Evaluation of Natural Language Processing

After the infamous ALPAC report in 1966, evaluation in NLP was considered a forbidden topic until the late 1980s in the USA. Around that time, a series of evaluation campaigns for speech processing and later speech understanding started occurring, with evaluation campaigns like TREC (Text

REtrieval Conference), which continue to this day. In Europe, similar evaluation events started a bit later, in the 1990s, with project such as Morpholympics (concerned with morphological tagging), GRACE (part-of-speech tagging) and Senseval (word sense disambiguation). The 2000s saw similar projects: Semeval (International Workshop on Semantic Evaluation), TECHNOLANGUE (A Permanent Evaluation and Information Infrastructure), EAGLES (Evaluation of Natural Language Processing Systems), and CLEF (Cross-Language Evaluation Forum) [4].

The 2010s saw even more natural language benchmarks, with projects such as SQuAD [5], SentEval [6], GLUE [7], SuperGLUE [8], XTREME [9], XGLUE [10], GENIE [11] and more, covered in more detail in Section 2.3. These benchmarks differ from their predecessors in that they provide more advanced datasets, offer online evaluation benchmarks (shared tasks), with some using novel methodologies of evaluation and model inspection, such as *adversarial filtering*, uploading and automatic evaluation of whole (e.g. transformer-based) models, crowd-sourcing of examples, explanation algorithms, visualizations, and more. [12].

2.2.1 Terminology

In most NLP environments (and, more generally, in machine learning), system development and evaluation requires partitioning available data (e.g. a corpus) into multiple disjoint subsets [13]:

- **Training data**, usually consisting of input vector (or scalar) and the corresponding output vector pairs (*labeled data*), often sourced from manual annotation. It can be used for training of supervised learning algorithms, but more broadly refers to any data used in the development of the NLP system.
- **Validation data (*development*, *dev*) data**, a labeled dataset of examples used to tune the hyperparameters of a model (e.g. the number of hidden layers in a neural network).

- **Development-test (*devtest*) data**, optionally used for testing a system's performance during development (self-testing with labeled data).
- **Test (*ground truth*, *reference*) data**, used in the evaluation of a system's performance.

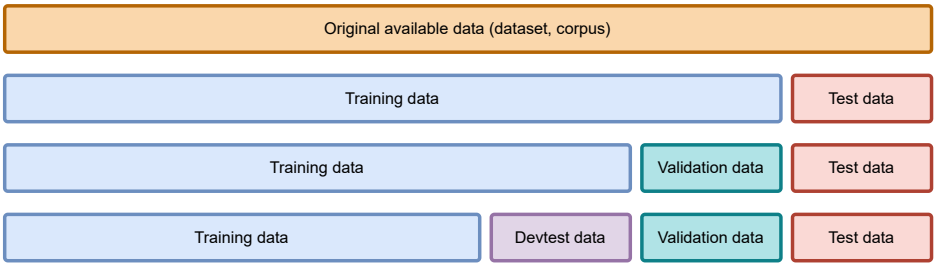

Figure 2.2: Possible dataset splits

Typical splits reserve as much data as possible for training and development. In the past, 70-20-10 percent splits used to be popular for training, hold-out (validation and optionally devtest) and test data respectively. This has changed in recent years, with the availability of massive (*big data*) datasets and deep learning models needing even more training data, shrinking the relative size of the test and validation sets, with modern typical splits of 98-1-1 or similar [13].

The need for disjoint subsets is crucial, as a fundamental principle of NLP (and more generally, machine learning) evaluation is that an evaluated system *must not be informed by test data*, as evaluations are intended to predict a system's performance on yet unseen data, i.e. the system's ability to generalise. Many benchmarks thus keep the test data labels (the output vectors of the input-output pairs) unavailable to the public, offering only evaluation on the hidden data in order to guarantee the validity of submitted results .

Today's NLP systems often consist of components, arranged in a processing pipeline. We can evaluate each component separately (component evaluation) or evaluate the entire pipeline (end-to-end evaluation), with both being

useful for different reasons - end-to-end evaluation offers a holistic quantification of a system's performance, focusing real-world tasks, while evaluation of individual components offers more understanding into a system's characteristics, e.g. by manipulating inputs to observe a syntactic parser's sensitivity in a controlled manner [13].

2.3 Existing Benchmarks

In recent years, starting in the late 2010s, numerous novel NLP benchmarks have surfaced. Many of these are sponsored or co-authored by large companies such as Google (GLUE, SuperGLUE, XTREME, GEM), Meta (Dynabench, SuperGLUE) and Microsoft (XGLUE). Besides innovations in dataset crafting, sourcing methodology and underlying evaluation metrics, modern benchmarks often offer web applications featuring public leaderboards along with result submission forms, so that anyone can submit results from their NLP model, creating a *real-time* shared task, updating every time someone makes a submission. The following section describes these tools and their contributions to the field of NLP benchmarking, partly focusing on their various innovations, but also on their public evaluation systems, the topic of which aligns closely with this thesis.

2.3.1 GLUE and SuperGLUE

The General Language Understanding Evaluation (GLUE) benchmark [7] from 2018 is one of the more influential NLP/NLU benchmarks in recent history. It consists of nine sentence or sentence-pair NLU tasks such as question answering, sentiment analysis and textual entailment. SuperGLUE [8] builds upon its predecessor with more difficult language understanding tasks and a software toolkit for evaluation.

The format of the GLUE/SuperGLUE benchmarks is model-agnostic, as any system which is able to process sentence and sentence pairs producing corresponding predictions is able to participate. Some of the benchmark's

tasks have limited training data available, as this encourages models with linguistic knowledge-sharing between tasks. In an interview conducted while researching their project for the purpose of this thesis, the authors noted they purposefully sourced relatively very large datasets in order to prevent easy manual annotation of test datasets.

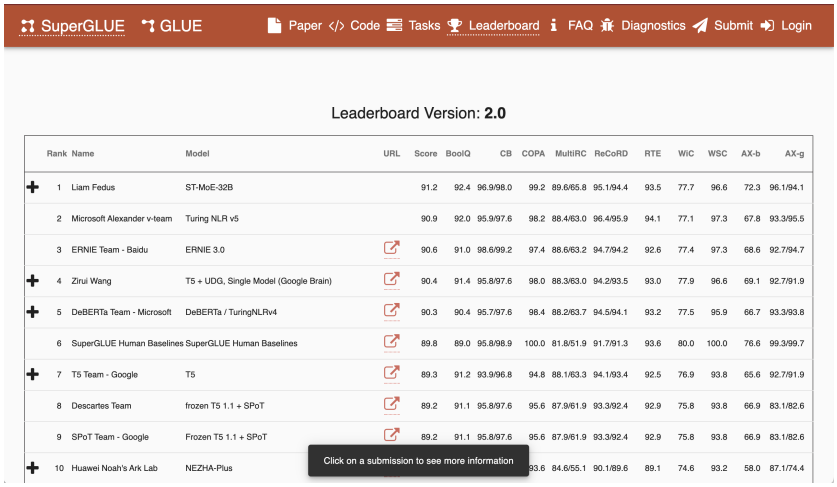

Figure 2.3: The SuperGLUE Benchmark leaderboard page

The benchmarks both provide online sites where users are able to submit the results from their systems for evaluation on the hidden test data. The results are then published in a couple of days onto the public leaderboard along with possible links to model code and research papers, some parameter information and a details sub-page for each model where one can inspect a breakdown of the score into individual metrics (e.g. a confusion matrix and Category-wise Matthew's Correlation Scores, a measure of the differences between actual values and predicted values) and diagnostics information from the diagnostic set, enabling more detailed linguistic analysis of models [14, 15].

SuperGLUE also introduced *jiant*, a software toolkit for natural language processing research based on PyTorch, made for conducting multitask and transfer learning experiments on English NLU tasks. It supports generating submission files for GLUE, SuperGLUE and EXTREME [16].

2.3.2 KLEJ

The KLEJ benchmark [17] (Kompleksowa Lista Ewaluacji Jezykowych) is a set of nine evaluation tasks for the Polish language understanding. Inspired by the GLUE benchmark (*klej* meaning *glue* in Polish), the benchmark consists of a diverse set of tasks, adopted from existing and novel datasets for named entity recognition, question-answering, textual entailment, and others. The authors also provide an automatic evaluation system and an online leaderboard to enable sharing model results, very similar in design and functionality to GLUE. The benchmark is model-agnostic, only requiring the preparation of a submission file in a specified format [18].

2.3.3 RussianSuperGLUE

RussianSuperGLUE [19] is another benchmark adhering to the GLUE and SuperGLUE methodology, marking the first time a complete test for the Russian language was developed, which is similar to its English analog. Many datasets were composed for the first time, and a public leaderboard of models with comparable results is also presented.

Besides evaluating the usual metrics for NLU and NLG tasks introduced by GLUE and SuperGLUE, the authors provide MOROCCO (*model resource consumption evaluation project*), aimed to evaluate RussianSuperGLUE model's performance inference speed and GPU RAM usage (temporal and spatial complexity). Users must submit their entire model as a Docker image besides the zip file containing evaluation results and in order to receive these additional measured model performance metrics on the public benchmark [20].

2.3.4 XTREME

The XTREME ((X) Cross-Lingual Transfer Evaluation of Multilingual Encoders) benchmark [9] is concerned with evaluating NLP over a diverse range of languages and tasks, encouraging more research on multilingual transfer

learning. It offers a web application with a public leaderboard, ranking the submissions based on their performance, with some additional data, such as links to the submission's related papers. The submission process is done through a Google Forms page using a zip file in the benchmark's specified format [21].

2.3.5 XGLUE

XGLUE [10] is a benchmark dataset to evaluate the performance of cross-lingual pre-trained models with respect to cross-lingual natural language understanding and generation. It is composed of 11 tasks spanning 19 languages, with the training data only available in English, with the intent of testing models for cross-lingual transfer capability, learning from English data and transferring what they learn to other languages.

The benchmark features a website with two static public leaderboards split by task types - natural language understanding and natural language generation. Submissions are accepted by emailing the authors prediction results generated on the test set in a specified format along with additional information such as model metadata, team/institution names and paper information. They also provide evaluation scripts in Python in a public Github repository [22].

2.3.6 SQuAD and SQuAD 2.0

SQuAD (Stanford Question Answering Dataset) is a reading comprehension dataset, consisting of crowd-sourced questions on a set of Wikipedia articles, where the answer to every question is a segment of text from the corresponding reading passage. SQuAD 2.0 also introduced questions where no answer is possible, requiring systems to determine such cases and abstain from answering [5, 23].

The benchmark is hosted online, providing the training and dev sets, Python-based evaluation scripts, sample prediction files and a public leader-

board with submitted results. Result submission is done by uploading a model's output results to a CodaLab worksheet, which is then evaluated by the authors on their hidden test set labels to preserve the integrity of test results [24].

2.3.7 GENIE

The GENIE [11] is another NLP leaderboard, innovating with *human-in-the-loop* evaluation, aiming to provide more accurate assessment of NLP progress using human evaluation of entries, gathered dynamically using crowd-sourcing (Amazon Mechanical Turk). The leaderboard summarizes the progress in text generation over a wide set of text generation tasks: question answering, summarization, machine translation and commonsense reasoning.

The benchmark provides separate leaderboards for its tasks with a table of public submissions showing performance on automatically evaluated metrics, such as BERTScore, ROUGE, METEOR and SacreBLEU, with a separate column for human evaluated performance of the models. Leaderboards also provide a chart of model performance on the task over time. The submissions are done by uploading predicted results onto the platform through a web form, along with some metadata [25].

2.3.8 GEM

GEM [26] is a benchmark environment for natural language generation (NLG), aiming to measure NLG progress across many tasks and languages, developing standards for evaluation of generated text using both automated and human metrics. GEM's web application provides two types of objects: *Task cards* and *Model cards*. Task cards provide the datasets along with their detailed documentation (source data, annotations and annotator information, social impact of the dataset, ...) for a specific NLP task, while Model cards provide a NLP model along with a description of the system, information about the training process, real-world use, dependencies, links to source

code and evaluation scores on automatic metrics [27].

2.3.9 Dynabench

Dynabench [28] is an open-source research platform for dynamic data collection and model benchmarking. It collects human-in-the-loop data dynamically - humans are tasked with finding adversarial examples that fool current state-of-the-art models, but not other humans. The authors argue that Dynabench addresses the fact that models quickly achieve *super-human* accuracy on other benchmarks, but fail on simple challenge examples and real-world scenarios. Its web application provides four natural language tasks, where users can create examples, validate them and submit models. Model submission is done using a tool called *Dynalab*, which helps users submit their entire models for dynamic evaluation on Dynabench [29].

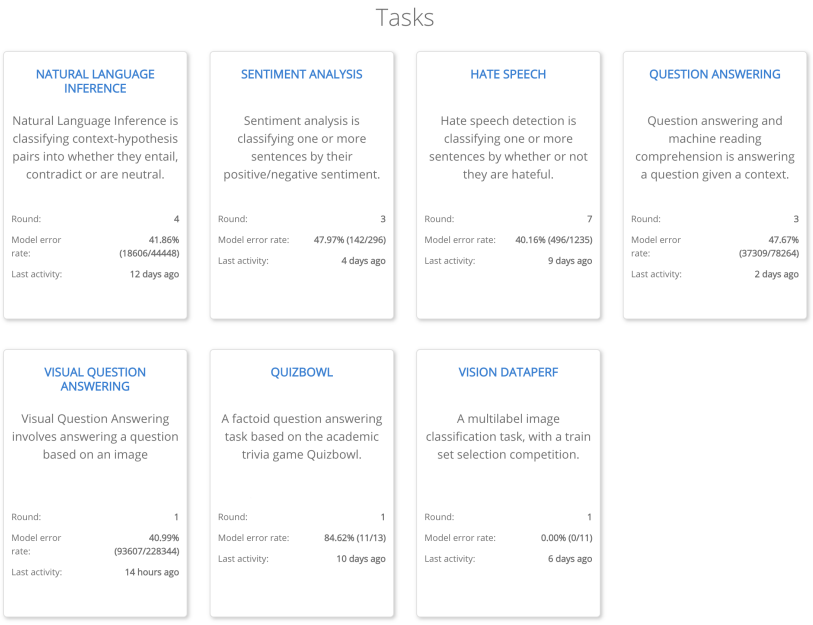

Figure 2.4: Dynabench tasks page, with links to leaderboards and some meta data about their usage

2.3.10 ExplainaBoard

ExplainaBoard is a benchmark that, in addition to providing the functionality of a standard NLP leaderboard, also allows researchers to diagnose strengths and weaknesses of a single system, interpret relationships between multiple systems and examine prediction results closely using the provided web application [30]. Its web application contains a table for each leaderboard, interactively displaying in-depth information (e.g. true and predicted labels for each entry) about a selected submission (model/system) or multiple submissions in case of pair-wise analysis and system combination [31].

Chapter 3

Architecture

The following chapter provides an architectural overview of the SloBench public benchmarking web platform and its evaluation framework. The first section introduces SloBench's web platform and provides an overview of its architecture. The second section is an in-depth look at the extendable open-source evaluation framework used in the scoring of user-uploaded submissions. Lastly, the third section describes the tools and frameworks (Django, React, Docker and Nginx) used to build the platform.

3.1 Overview

The SloBench platform is a web application consisting of a React based frontend and Django backend, providing its users with different functionality based on their roles. Figure 3.1 presents an overview of its architecture. The presentation layer contains the platform's main functionality – the creation of task-specific NLP leaderboards, where Editors provide datasets and evaluation criteria in the form of an evaluation Docker image. Contributors can then download the unlabeled test data for a specific task, generate predicted labels using their systems, and submit results to the leaderboard for automatic evaluation using the SloBench evaluation framework. The service layer coordinates the application's task specific evaluation processes, send-

ing of emails (upon submission evaluation completion and weekly submission aggregates) and data storage. The user roles and their functionality (Figure 3.1's presentation layer) are described in more detail in Chapter 4, while the service layer and its underlying technologies are examined in more detail in Sections 3.2 and 3.3.

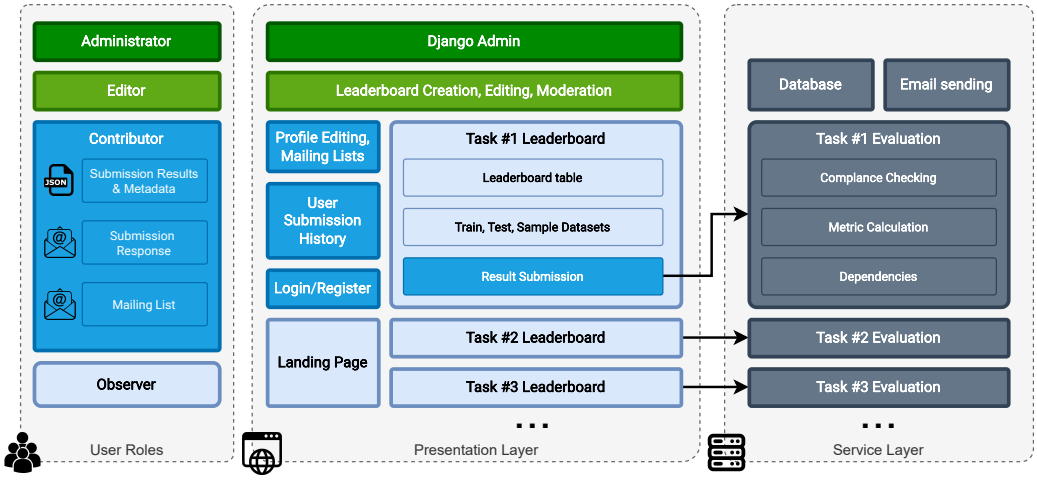

Figure 3.1: High level architecture of the Slobench platform, depicting its user roles, presentation, and service layers

3.2 Evaluation Images and Scripts

SloBench provides source code for the evaluation of each of its natural language processing tasks, with the aim of transparency, repeatably and enabling local testing of one's results before uploading them for evaluation and display on a public leaderboard. The evaluation scripts are published in a publicly accessible Git repository[1]. The scripts are also built and published as Docker images, which enables the running of pseudo-sandboxed evaluation processes on the SloBench server with virtually unbounded environment setups. This means evaluation scripts can be written in any programming

[1]SloBench project's evaluation scripts: www.github.com/clarinsi/SloBench-eval-docker

language and use any dependencies that can run in a Docker container. Currently, all of the existing evaluation images are Python based, as there is a number of existing libraries implementing popular evaluation techniques (e.g. *scikit-learn*'s metrics module for accuracy and F1 metrics, *pyrouge* - a Python wrapper for the ROUGE summarization evaluation package, ...), but SloBench's evaluation framework places no limitations on the evaluation dependencies and input file types.

All evaluation images follow the same basic architecture as seen in Figure 3.2. The *run.py* script, being identical across all of the evaluation images, contains the functions for running individual evaluation scripts in *evaluate.py*, handling of input files along with their decompression and the creation of a TSEO - Task Submission Evaluation Object (Figure 3.3) that contains the computed metrics from the *evaluate.py* script along with information about the length of the evaluation and any possible errors that might have occurred during the process. Figure 3.4 describes how the evaluation of a task is executed in the backend after a user successfully submits some results on the SloBench site.

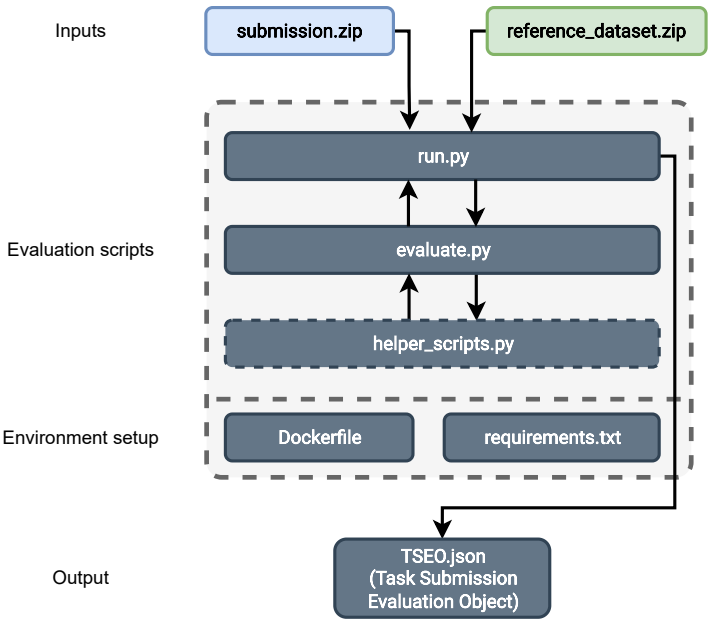

Figure 3.2: Architecture of an evaluation script Docker image.

```
{
  "status": "S",
  "metrics": {
    "Precision": 0.9811290281848772,
    "Recall": 0.9829913004684363,
    "F1 score": 0.9818619988615366
  },
  "evaluation_time": 0.459997,
  "error_report": ""
}
```


Figure 3.3: Example of a successful TSEO (Task Submission Evaluation Object), returned by an evaluation container after computing the metrics for a given task.

3.2.1 Creating an evaluation script

Creating an evaluation script is a required step in creating a new leaderboard on SloBench. One must first define the metrics they wish to use in the evaluation of a task, along with providing sample submission and reference data (usually compressed, zip format) files. When creating a new script, it is best to extend an existing evaluation script along with its dependencies, test locally and then create a pull request on the public repo which hosts the existing evaluation scripts. The further process of creating a leaderboard on the SloBench web platform is described in more detail in Chapter 4.

Some key factors to take into account when creating a new evaluation task for SloBench are: the sufficient size of reference datasets - they should be large enough as to prevent simple manual annotation, the usage of previously unseen labeled data - ensuring valid results of the evaluation on truly hidden data that an NLP model has not been *taught* on. More best practices on

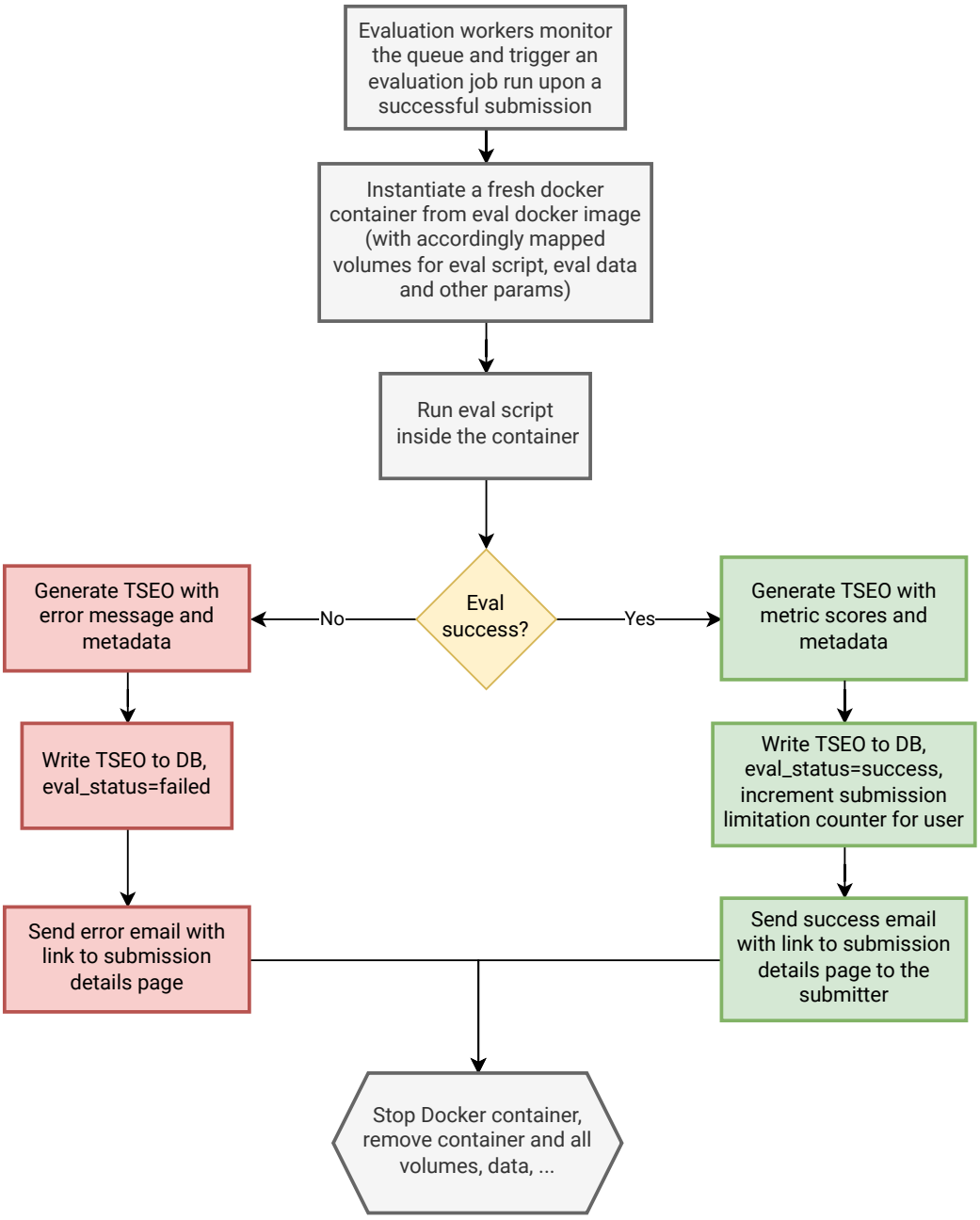

Figure 3.4: Flow chart of the evaluation process.

crafting NLP evaluation datasets are described in [12].

3.3 Tools and Libraries

Django

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. It offers features like an object-relational mapper in which you describe your database layout in Python code, a user authentication system with built-in security, an automatic admin interface which can be used out-of-the-box as an internal management tool and a rich ecosystem of external packages which extend Django's functionality [32].

React

React is a JavaScript library for building user interfaces based on UI components. It is commonly used as a frontend base in the development of single-page applications [33]. Because of it's popularity, many design systems, such as IBM's Carbon Design System [34] (the base for CJVT's design system, used in the SloBench project) offer prebuilt customizable components which speed up development significantly.

Docker

Docker is an open platform for developing, shipping and running applications in containers — standardized executable components, combining application source code with the operating system (OS) libraries and dependencies required to run that code in any environment [35]. It enables the packaging of the entire SloBench web application (frontend, backend, evaluation workers and Nginx reverse-proxy) into a simple-to-run multi-container application with Docker Compose and the packaging of benchmark-specific evaluation scripts into reusable evaluation containers, which enable local execution (testing) of evaluation in the exact manner it is then performed on the SloBench server.

Nginx

Nginx is an open source web server that can also be used as a reverse proxy, load balancer, mail proxy and HTTP cache [36]. SloBench uses the service as a reverse-proxy and static-file server, providing additional security for sensitive media files (i.e. uploaded results and hidden test dataset labels).

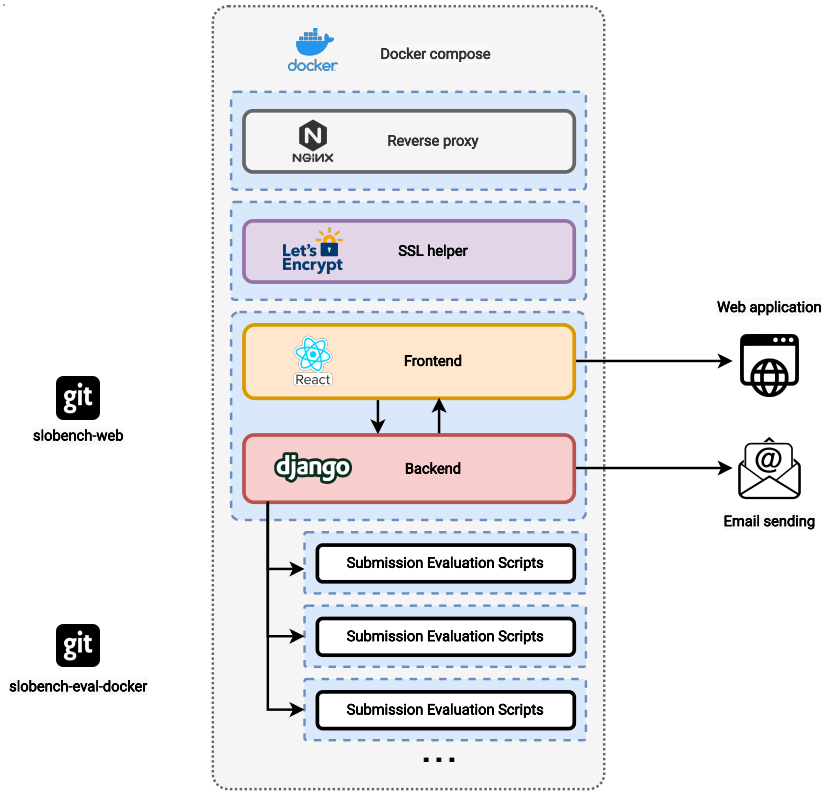

Figure 3.5: The Slobench platform is hosted by four Docker containers, with additional containers started upon submission evaluation.

Chapter 4

Functionality and Workflows

This chapter presents the main functionality of the SloBench web platform, along with its four user roles (observer, contributor, editor and administrator) and their workflows (leaderboard creation, result submission and leaderboard moderation).

The SloBench service is a modern web application created using Django for its backend and React with a custom UI library for its frontend. Upon visiting the site, the user sees the landing page, displaying some basic information about the project ("About" section), frequently asked questions ("F.A.Q." section") and a list of leaderboards displayed as cards with the leaderboard title, short description and submission counters. The application supports four user roles:

- **Observer** (Figure 4.2), the anonymous user role, which is able to view public leaderboards along with their submission results and registering or logging in into the site.
- **Contributor** (Figure 4.3), the basic registered and logged in user role, able to submit their results to a specific leaderboard for evaluation, view their previous submissions and edit their profile, along with inheriting the Observer role's functionality.
- **Editor** (Figure 4.4), the elevated user, is able to edit and moderate

the leaderboards of which they are assigned to as "editors", viewing the users which have submitted to their leaderboards, along with inheriting the Contributor role's functionality.

- **Admin** (Figure 4.5), the super-user role, is able to access the Django Admin interface, view and manage all registered users on the site, view and edit all public and private leaderboards, and create new leaderboards, along with inheriting the Editor role's functionality.

The SloBench web application provides leaderboard creation and editing capabilities, described in Section 4.1, along with moderation capabilities, described in Section 4.3 to editors and admins, while providing result submission capabilities, described in Section 4.2 to contributors.

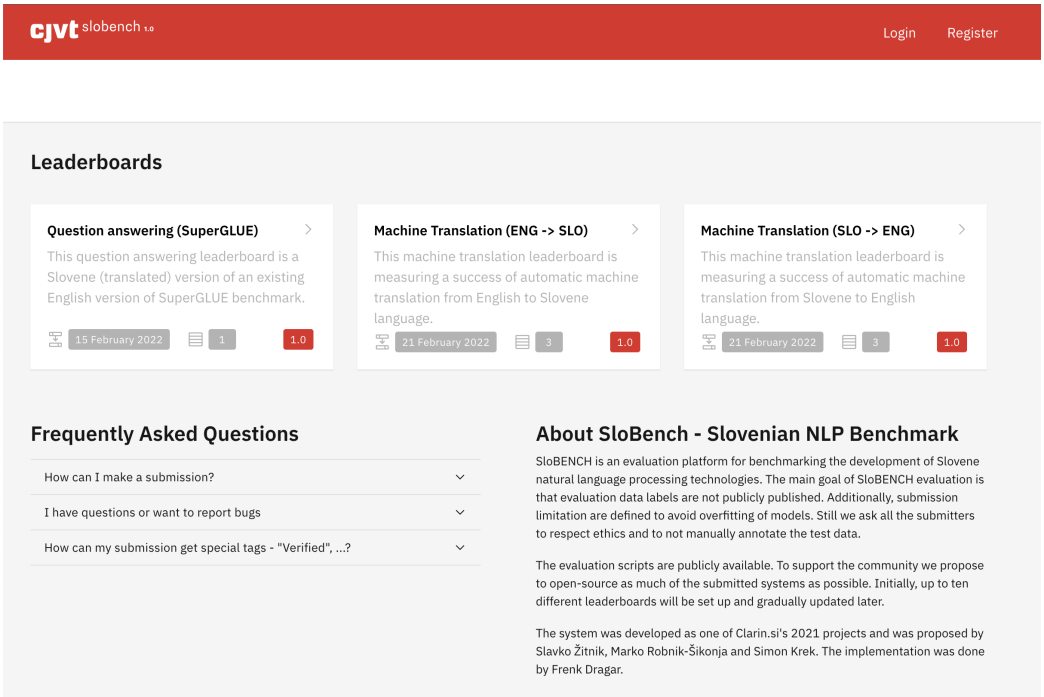

Figure 4.1: The landing page contains links to leaderboards in the form of leaderboard cards, Frequently Asked Questions (FAQ) and an About section

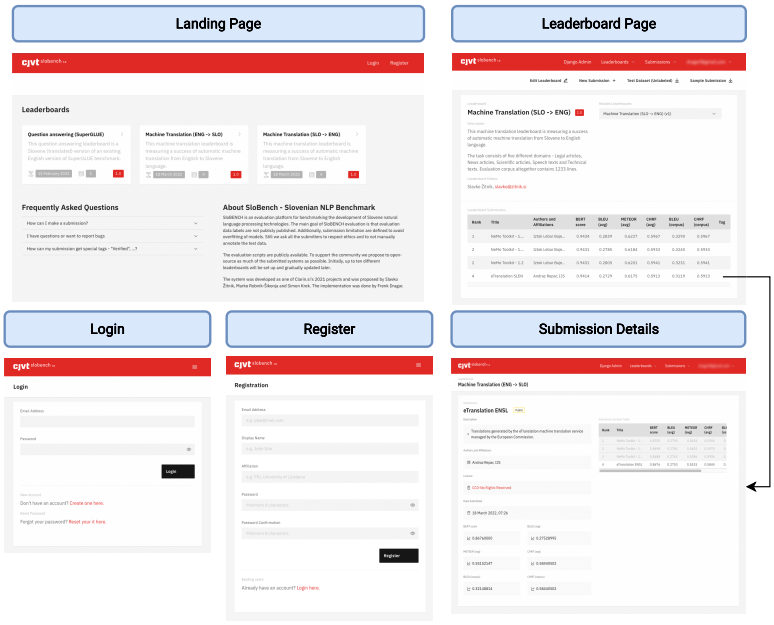

Figure 4.2: Views, available to the Observer user role

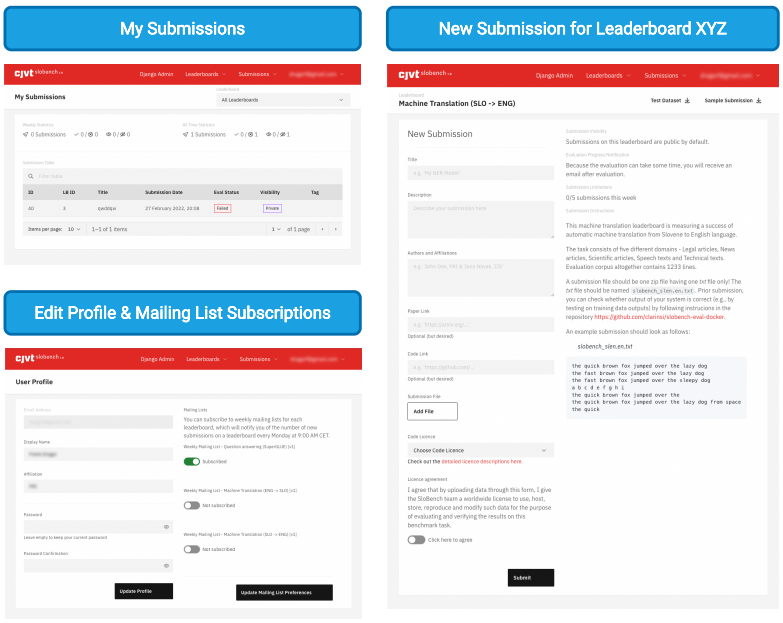

Figure 4.3: Views, available to the Contributor user role

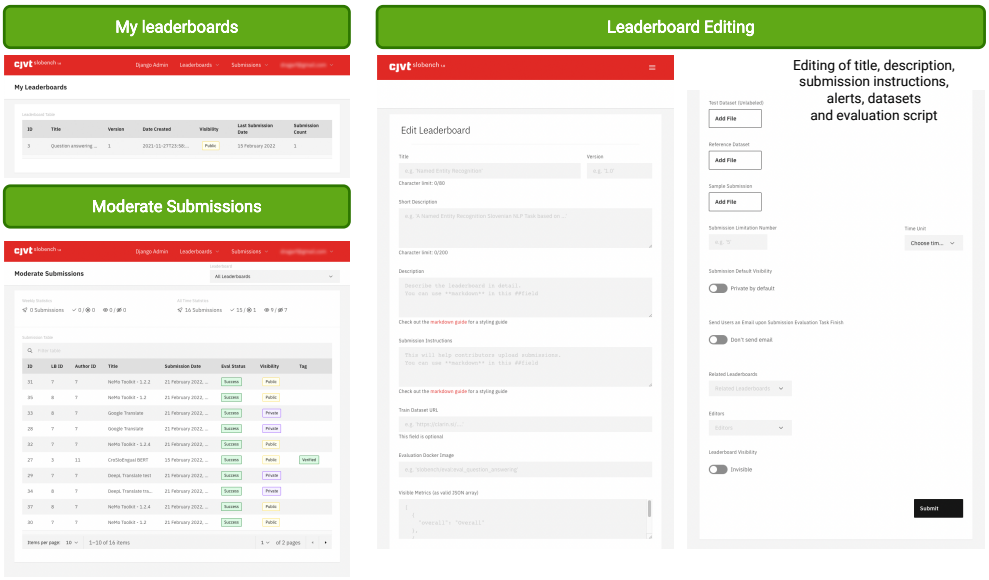

Figure 4.4: Views, available to the Editor user role

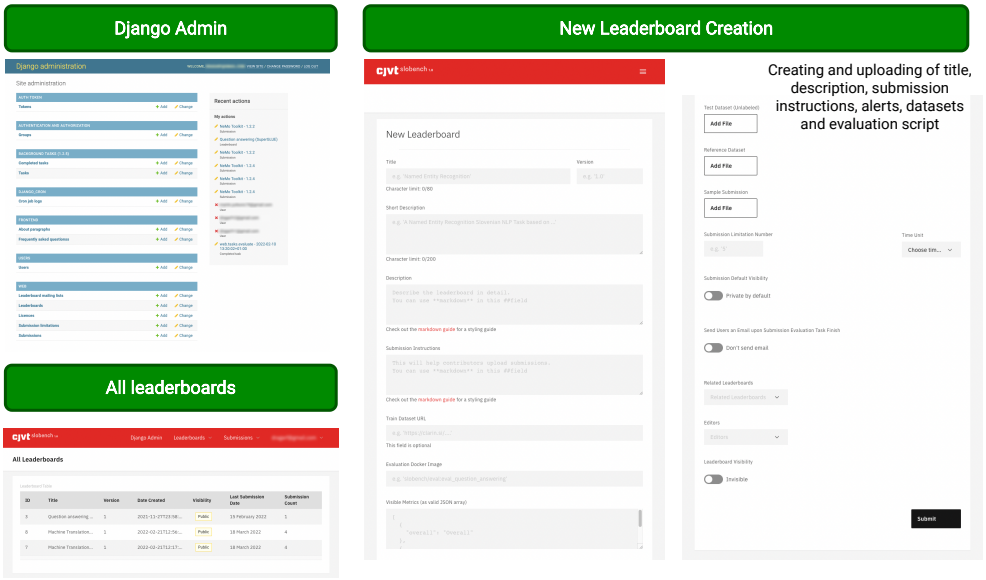

Figure 4.5: Views, available to the Admin user role

4.1 Leaderboard Creation

A leaderboard can be created on the site by navigating to "Leaderboard → New Leaderboard" on the top of the navigation bar for Admin role users. The leaderboard creation workflow is illustrated in Figure 4.7. The leaderboard creator must provide the leaderboard *title* (e.g. "Named Entity Recognition") and *version* (e.g. "1.0") - useful for updating datasets and evaluation scripts, but not deleting old ones, a *short description*, which is seen in the leaderboard card on the landing page, a *long description* which supports markdown syntax and is seen on the leaderboard main page, *submission instructions*, seen on the "New Submission" page for this leaderboard, the *train dataset URL*, which can be e.g. a link to a CLARIN repository hosting the dataset, the *evaluation Docker image*, which is the Docker repository with a specific tag corresponding to the previously built evaluation script (e.g. "slobench/eval:qa-1.0"), described in detail in Chapter 3, the *visible metrics*, a JSON array of key-value (TSEO name metric – display metric name) pairs, which dictate the order of metrics in the table display of submission results, with the first metric in the array providing the default sorting key. Then, one must upload the *test* and *reference* datasets and a *sample submission*, usually zip file. The reference dataset and sample submission files are available for all users to download on the leaderboard and leaderboard submission page, while the reference dataset is a protected asset, only available through the leaderboard edit page or Django admin interface.

With the aim of preventing *benchmark overfitting* of the models, caused by too frequent submissions on a particular leaderboard, a leaderboard creator can place a *submission limitation number* along with a *time unit* (day, week, month), e.g. "5 submissions per week", prohibiting a particular user from submitting more than a specified number of submissions per time unit. This also provides the mechanism for disallowing submissions on a leaderboard, simply by putting the number 0 as the number of allowed submissions. Other customisable options include the ability to make all *submissions visible or hidden by default* and *sending users an email upon finishing the evalua-*

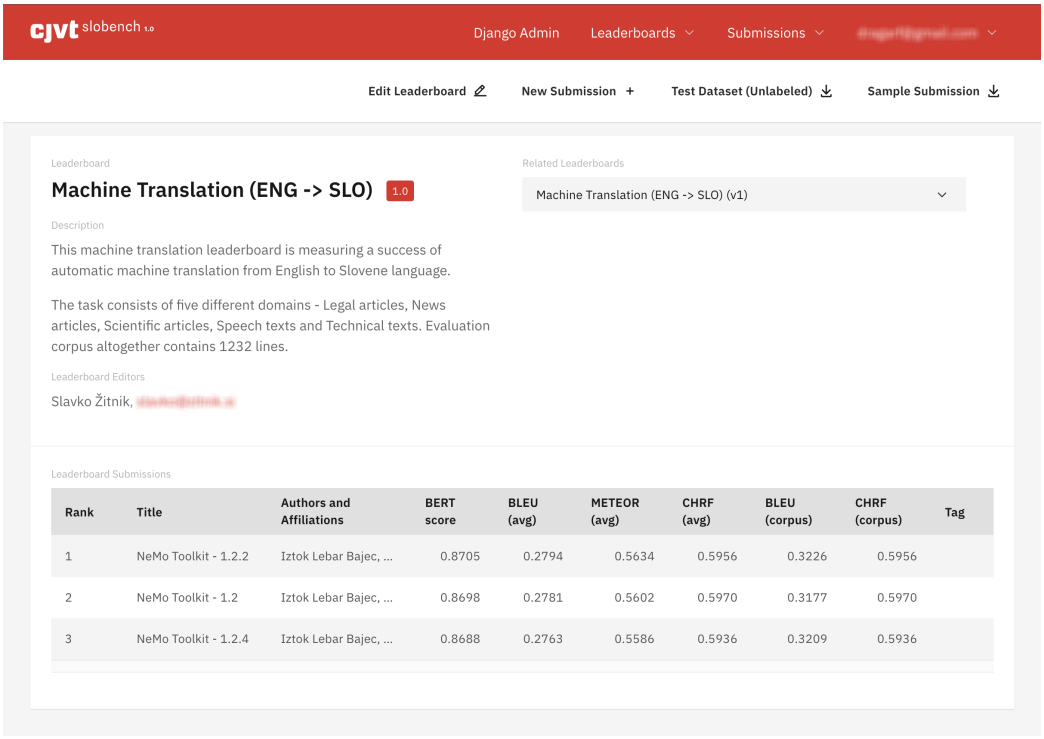

Figure 4.6: Screenshot of the Machine Translation task leaderboard page

tion job, useful for notifying users of their score if a particular task evaluation job is known to run for a longer period of time. A leaderboard author can also choose to provide *related leaderboards*, e.g. linking previous versions of a leaderboard or linked translation tasks for quick access on a leaderboard page - especially useful for leaderboard versioning, as one can link even hidden leaderboards - making public only the most recent one, but providing access to previous versions with the related leaderboards drop-down menu on the leaderboard page. Finally, one can assign *editors* to a leaderboard, enabling them to edit the leaderboard and moderate its submissions, and choose between the leaderboard being visible or hidden from the landing page (but still accessible with a direct link, to support leaderboard versioning).

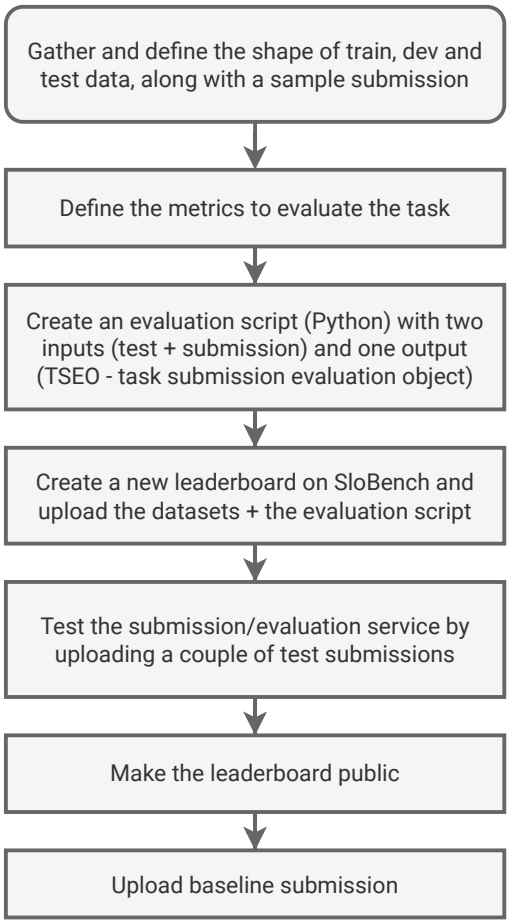

Figure 4.7: Leaderboard Creation Workflow

4.2 Result Submission

A registered and logged in user can submit their results on a particular leaderboard by going to the leaderboard page and clicking the "New Submission" link at the top of the page. The process is described in Figure 4.8. They are taken to a new page with a submission form on the left and information about submission default visibility, evaluation progress notification (by email or on the page) and submission limitations (e.g. 3/5 submission per week thus far) and submission instructions on the right. The user can download the train and test datasets along with the sample submission on this page.

The submission author must provide submission results which is a zip

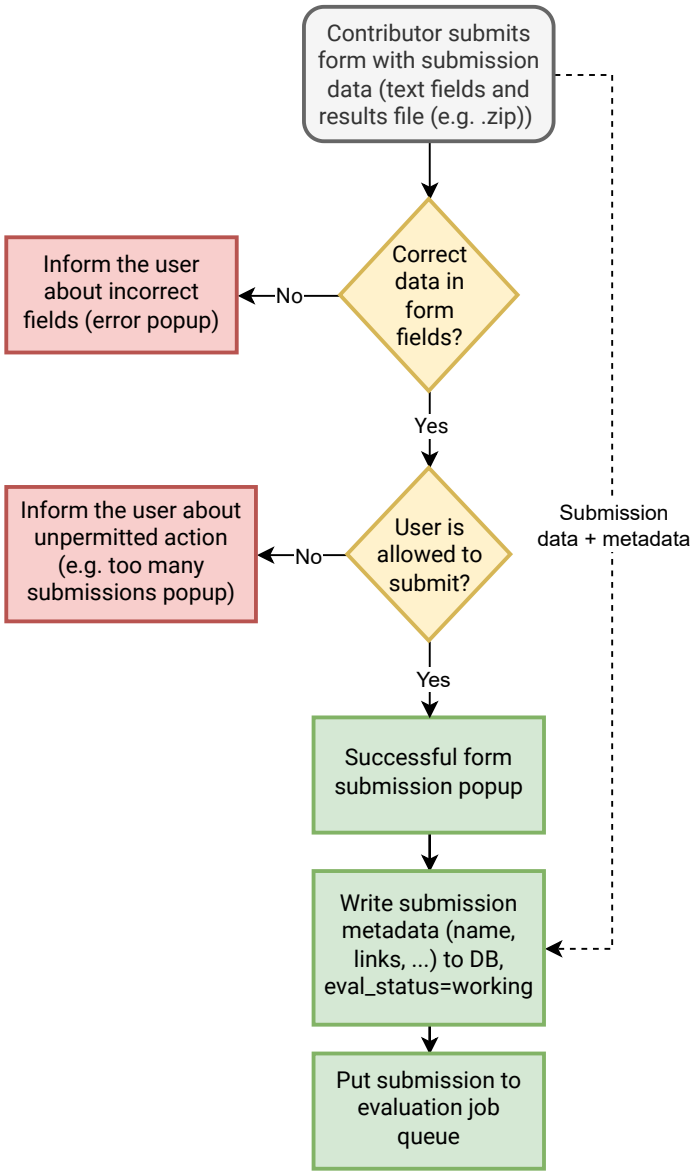

Figure 4.8: Flow chart of the submission process

file for most leaderboards, the *title* and *description* of the system used to create the submission, along with the system's *authors and affiliations* (e.g. Frenk Dragar, FRI and Slavko Žitnik, FRI) - this field is not connected to the "Affiliation" field in a user profile, as an author can be affiliated with multiple organisations and can pick the relevant one per submission. Authors

can optionally provide *paper* and *code links*, which is desired by the SloBench project with the aim of soliciting transparent and repeatable results. The authors must choose the licence under which the model is published and agree to the licence agreement, which gives the SloBench team a worldwide license to use, host, store, reproduce and modify such data for the purpose of evaluating and verifying the results on this benchmark task.

After passing the basic form validation and successfully submitting, the user is taken to a new page - the "Submission Detail" page. If the submission evaluation is successful, they are shown the performance of their system against the hidden test set data. Otherwise, in the event of evaluation failure, the page shows a detailed trace of the evaluation failure, helping the user diagnose what went wrong, e.g. "Submission and Reference set label count mismatch".

4.3 Leaderboard Moderation

SloBench editors and admins are able to moderate the uploaded submissions by changing their visibility between public and private, marking a submission as *baseline*, thus adding a special "Baseline" tag to the submission in all visible tables on the site, marking a submission as *verified*, indicating the submission has been checked for repeatably and the results have been confirmed as valid, along with other possible criteria defined per leaderboard. Some admin user features are not available on the SloBench frontend, but instead accessed through the Django Admin interface in the menu bar. Through Django Admin, an admin can trigger the re-evaluation of submissions, edit "about" and "frequently asked questions" sections on the landing page, manage registered users, and more, courtesy of the standard features inherited from Django Admin.

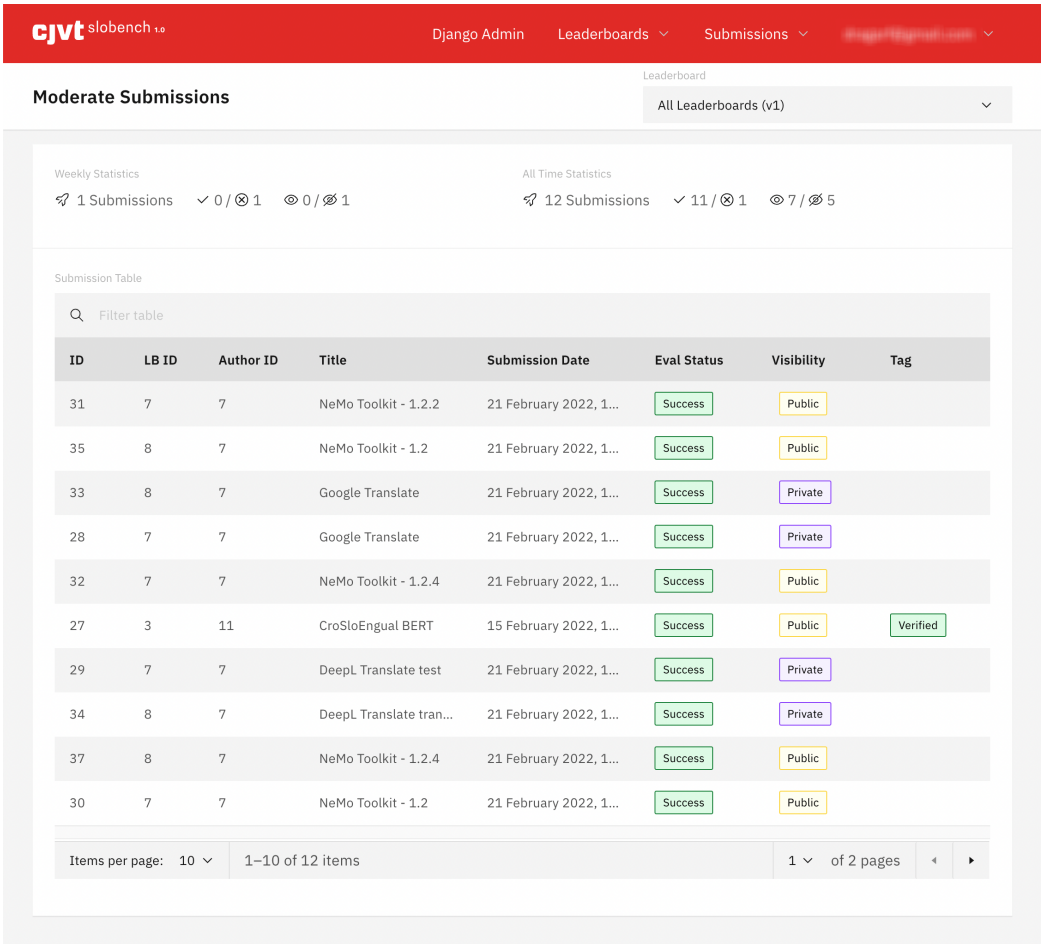

Figure 4.9: The moderation page provides an overview of submissions on the platform's leaderboards

Chapter 5

Evaluation

Natural language processing benchmark popularity has grown immensely in recent years. We provide an analysis of these benchmarks in Chapter 2. This chapter compares their similarities and differences with the SloBench platform.

All of the mentioned benchmarks in Chapter 2 provide an online leaderboard with submitted results in a table of achieved scores on various metrics. Most of these also provide a web form to enable submission of results onto the leaderboard, while some opt for sending an email with included results and some related metadata to the leaderboard authors.

Evaluation of submissions is presumably done manually by the authors of a specific leaderboard for most of the inspected leaderboards, as their submission description pages state that results are publicly available in a couple of business days after submission. Only GENIE and Dynabench state that their result evaluation is automatic upon submission. All of the benchmarks provide source code to their evaluation scripts, largely written in Python. Some benchmarks provide more advanced tools, such as *Jiant*, a wrapper library for transformer-based models, which support exporting results for GLUE, SuperGLUE and XTREME, or RussianSuperGLUE's MOROCCO, which enables the calculation of a model's inference speed and GPU RAM usage. All benchmarks are model-agnostic, meaning they are able to eval-

uate a submission, regardless of the underlying model architecture. This is achieved by requiring the evaluated systems to produce results in a required output format, most commonly a compressed zip file with .json, .jsonl or .txt files inside. Some even support the uploading of whole models for evaluation on their web service.

Most of the benchmarks on the list are multi-task, with the exception of SQuAD, which is dataset with an included leaderboard. They differ however, in their treatment of the tasks - some provide only one leaderboard with the separate tasks (or rather, the scores for those tasks) representing columns in a table - this is the style of GLUE and its clones. Others present multiple different leaderboards on the same platform, with each task (or multiple tasks) representing an individual table on their web application. Here, columns are commonly different metrics that have been calculated on same task results.

Some of the most recent benchmarks provide advanced functionality - Explainaboard aims to provide detailed information regarding the submitted systems (e.g. fine-grained results, confidence intervals) and enable the *explainability* of a particular model's results and additionally provide exploration of systems' advantages or weaknesses over others by side-to-side comparison over various metrics. Dynabench collects its data dynamically, with humans and models in the loop together, aiming to succeed *static* benchmarks and their problems like models achieving super-human results quickly, when in reality they only overfit to the benchmark and fail to perform on real-world tests. While all of these benchmarks can of course be supplemented with additional tasks and metrics, we label the latter group as *extendable* because of the possibility of adding more tasks and categories to their platforms while keeping the structure of existing leaderboard pages.

SloBench aims to differentiate itself from other projects by providing an extendable evaluation framework based on Docker images that are able to run on a web service and automatically evaluate submissions as soon as they are submitted. We also provide an extendable web platform which enables the dynamic creation new leaderboards, their editing and moderation.

	GLUE/SuperGLUE	KLEJ	RussianSuperGLUE	XTREME	SQuAD	GENIE	GEM	Dynabench	Explainaboard	SloBench
Online leaderboard	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Automatic (online) evaluation						✓		✓		✓
Model-agnostic	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Hidden test dataset labels	✓	✓		✓	✓	✓		✓	✓	✓
Public evaluation scripts	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Number of tasks	10	9	9	4	1	4	13	7	13	10[1]
Extendable						✓	✓	✓	✓	✓
Advanced analysis									✓	
Human-in-the-loop						✓		✓		

Table 5.1: Comparison of related works - public NLP benchmarks and the SloBench benchmark

[1]Three tasks are already available, with seven more evaluation scripts prepared - the tasks will be published as soon as their datasets are prepared

Chapter 6

Conclusion

This thesis explores recent efforts in NLP benchmarking with the intent of creating an automated evaluation and public benchmarking system for the Slovene natural language processing community. We first propose an extendable evaluation framework that is capable of supporting a broad range of evaluation tasks. These can have completely different dependencies and requirements, along with arbitrary data input formats. Chapter 3 also provides the instructions on how to prepare new evaluation scripts in the format supported by SloBench and an explanation of the evaluation process on the SloBench web service.

Afterwards, in Chapter 4 we explore the functionality of the SloBench web service, inspecting the four user types of the application and provide detailed descriptions of the leaderboard creation, submission and moderation processes. These provide the documentation of the SloBench web application, giving some insight into the design of the platform.

At the time of writing, the platform hosts the first three NLP task and their public leaderboards - Question Answering, Machine Translation (English to Slovene) and Machine Translation (Slovene to English). The first task is a Slovene translated version of the SuperGLUE benchmark, consisting of 6 sub-tasks, while the second and third are automatic translation tasks on five different domains - Legal articles, News articles, Scientific articles, Speech

texts and Technical texts. Seven more tasks are scheduled to be published on the service shortly, with the general and extensible service enabling the creation of any future task leaderboard, pending the collection of datasets and formalisation of evaluation metrics in evaluation Docker images.

The web service provides the basic functionality required for a public automated benchmark in any language or tasks. Some very recent benchmark systems (e.g. Dynabench, Explainaboard) experiment with features such as human-in-the-loop data collection and evaluation, novel methods for the comparison and analysis of submitted systems, visualisation methods for describing performance, evaluating the efficiency of systems and more. These solutions are not yet standard and don't necessarily provide more analytical value. Should they prove useful however, the evaluation framework and public web application are both able of extensions and provide the option to add such new features if any needs or ideas arise.

Bibliography

[1] Prakash M Nadkarni, Lucila Ohno-Machado, and Wendy W Chapman. "Natural language processing: an introduction". In: *Journal of the American Medical Informatics Association* 18.5 (2011), pp. 544–551.

[2] Margaret King. "When is the next Alpac report due?" In: *10th International Conference on Computational Linguistics and 22nd Annual Meeting of the Association for Computational Linguistics*. 1984, pp. 352–353.

[3] Elizabeth D Liddy. "Natural language processing". In: *Encyclopedia of Library and Information Science, 2nd Ed.* (2001).

[4] Patrick Paroubek, Stéphane Chaudiron, and Lynette Hirschman. "Principles of evaluation in natural language processing". In: *Revue TAL* 48.1 (2007), pp. 7–31.

[5] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. *SQuAD: 100,000+ Questions for Machine Comprehension of Text*. 2016. arXiv: 1606.05250 [cs.CL].

[6] Alexis Conneau and Douwe Kiela. *SentEval: An Evaluation Toolkit for Universal Sentence Representations*. 2018. arXiv: 1803.05449 [cs.CL].

[7] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. *GLUE: A Multi-Task Benchmark and Analy-*

sis Platform for Natural Language Understanding. 2019. arXiv: 1804.07461 [cs.CL].

[8] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. *SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems*. 2020. arXiv: 1905.00537 [cs.CL].

[9] Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. *XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalization*. 2020. arXiv: 2003.11080 [cs.CL].

[10] Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, Xiaodong Fan, Ruofei Zhang, Rahul Agrawal, Edward Cui, Sining Wei, Taroon Bharti, Ying Qiao, Jiun-Hung Chen, Winnie Wu, Shuguang Liu, Fan Yang, Daniel Campos, Rangan Majumder, and Ming Zhou. *XGLUE: A New Benchmark Dataset for Cross-lingual Pre-training, Understanding and Generation*. 2020. arXiv: 2004.01401 [cs.CL].

[11] Daniel Khashabi, Gabriel Stanovsky, Jonathan Bragg, Nicholas Lourie, Jungo Kasai, Yejin Choi, Noah A. Smith, and Daniel S. Weld. *GENIE: A Leaderboard for Human-in-the-Loop Evaluation of Text Generation*. 2021. arXiv: 2101.06561 [cs.CL].

[12] Samuel R. Bowman and George E. Dahl. *What Will it Take to Fix Benchmarking in Natural Language Understanding?* 2021. arXiv: 2104.02145 [cs.CL].

[13] Philip Resnik and Jimmy Lin. "Evaluation of NLP systems". In: *The handbook of computational linguistics and natural language processing* 57 (2010).

[14] *General Language Understanding Evaluation (GLUE) benchmark*. URL: https://gluebenchmark.com/ (visited on 02/15/2022).

[15] *SuperGLUE Benchmark*. URL: https://super.gluebenchmark.com/ (visited on 02/15/2022).

[16] Jason Phang, Phil Yeres, Jesse Swanson, Haokun Liu, Ian F. Tenney, Phu Mon Htut, Clara Vania, Alex Wang, and Samuel R. Bowman. *jiant 2.0: A software toolkit for research on general-purpose text understanding models*. http://jiant.info/. 2020.

[17] Piotr Rybak, Robert Mroczkowski, Janusz Tracz, and Ireneusz Gawlik. *KLEJ: Comprehensive Benchmark for Polish Language Understanding*. 2020. arXiv: 2005.00630 [cs.CL].

[18] *KLEJ Benchmark*. URL: https://klejbenchmark.com/ (visited on 02/15/2022).

[19] Tatiana Shavrina, Alena Fenogenova, Anton Emelyanov, Denis Shevelev, Ekaterina Artemova, Valentin Malykh, Vladislav Mikhailov, Maria Tikhonova, Andrey Chertok, and Andrey Evlampiev. "RussianSuperGLUE: A Russian Language Understanding Evaluation Benchmark". In: *arXiv preprint arXiv:2010.15925* (2020).

[20] *Russian SuperGLUE Benchmark*. URL: https://russiansuperglue.com/ (visited on 02/15/2022).

[21] *XTREME - Cross-Lingual Transfer Evaluation of Multilingual Encoders*. URL: https://sites.research.google/xtreme (visited on 02/15/2022).

[22] *XGLUE - Cross-lingual GLUE*. URL: https://microsoft.github.io/XGLUE/ (visited on 02/15/2022).

[23] Pranav Rajpurkar, Robin Jia, and Percy Liang. *Know What You Don't Know: Unanswerable Questions for SQuAD*. 2018. arXiv: 1806.03822 [cs.CL].

[24] *SQuAD - The Stanford Question Answering Dataset*. URL: https://rajpurkar.github.io/SQuAD-explorer/ (visited on 02/15/2022).

[25] *GENIE, a leaderboard for natural language generation tasks*. URL: https://genie.apps.allenai.org/ (visited on 02/15/2022).

[26] Sebastian Gehrmann, Tosin Adewumi, Karmanya Aggarwal, Pawan Sasanka Ammanamanchi, Aremu Anuoluwapo, Antoine Bosselut, Khyathi Raghavi Chandu, Miruna Clinciu, Dipanjan Das, Kaustubh D. Dhole, Wanyu Du, Esin Durmus, Ondřej Dušek, Chris Emezue, Varun Gangal, Cristina Garbacea, Tatsunori Hashimoto, Yufang Hou, Yacine Jernite, Harsh Jhamtani, Yangfeng Ji, Shailza Jolly, Mihir Kale, Dhruv Kumar, Faisal Ladhak, Aman Madaan, Mounica Maddela, Khyati Mahajan, Saad Mahamood, Bodhisattwa Prasad Majumder, Pedro Henrique Martins, Angelina McMillan-Major, Simon Mille, Emiel van Miltenburg, Moin Nadeem, Shashi Narayan, Vitaly Nikolaev, Rubungo Andre Niyongabo, Salomey Osei, Ankur Parikh, Laura Perez-Beltrachini, Niranjan Ramesh Rao, Vikas Raunak, Juan Diego Rodriguez, Sashank Santhanam, João Sedoc, Thibault Sellam, Samira Shaikh, Anastasia Shimorina, Marco Antonio Sobrevilla Cabezudo, Hendrik Strobelt, Nishant Subramani, Wei Xu, Diyi Yang, Akhila Yerukola, and Jiawei Zhou. *The GEM Benchmark: Natural Language Generation, its Evaluation and Metrics*. 2021. arXiv: 2102.01672 [cs.CL].

[27] *GEM Benchmark*. URL: https://gem-benchmark.com/ (visited on 02/15/2022).

[28] Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. *Dynabench: Rethinking Benchmarking in NLP*. 2021. arXiv: 2104.14337 [cs.CL].

[29] *Dynabench*. URL: https://dynabench.org/ (visited on 02/15/2022).

[30] Pengfei Liu, Jinlan Fu, Yang Xiao, Weizhe Yuan, Shuaicheng Chang, Junqi Dai, Yixin Liu, Zihuiwen Ye, Zi-Yi Dou, and Graham Neubig. *Ex-*

plainaBoard: An Explainable Leaderboard for NLP. 2021. arXiv: 2104.06387 [cs.CL].

[31] *ExplainaBoard - An Explainable Leaderboard for NLP*. URL: http://explainaboard.nlpedia.ai/ (visited on 02/15/2022).

[32] *Django - a high-level Python web framework*. URL: https://www.djangoproject.com/ (visited on 02/14/2022).

[33] *React - A JavaScript library for building user interfaces*. URL: https://reactjs.org/ (visited on 02/14/2022).

[34] *Carbon Design System - IBM's open source design system for products and digital experiences*. URL: https://www.carbondesignsystem.com/ (visited on 02/14/2022).

[35] *Docker - an open platform for developing, shipping, and running applications*. URL: https://www.docker.com/ (visited on 02/14/2022).

[36] *NGINX - Advanced Load Balancer, Web Server and Reverse Proxy*. URL: https://www.nginx.com/ (visited on 02/14/2022).