UniHD at TSAR-2022 Shared Task: Is "Compute All We Need"' for Lexical Simplification?

Dennis Aumiller and **Michael Gertz**
Institute of Computer Science
Heidelberg University
{aumiller, gertz}@informatik.uni-heidelberg.de

Abstract

Previous state-of-the-art models for lexical simplification consist of complex pipelines with several components, each of which requires deep technical knowledge and fine-tuned interaction to achieve its full potential. As an alternative, we describe a frustratingly simple pipeline based on prompted GPT-3 responses, beating competing approaches by a wide margin in settings with few training instances. Our best-performing submission to the English language track of the TSAR-2022 shared task consists of an "ensemble" of six different prompt templates with varying context levels. As a late-breaking result, we further detail a language transfer technique that allows simplification in languages other than English. Applied to the Spanish and Portuguese subset, we achieve state-of-the-art results with only minor modification to the original prompts. Aside from detailing the implementation and setup, we spend the remainder of this work discussing the particularities of prompting and implications for future work. Code for the experiments is available online.[1]

1 Introduction

With recent advancements in Machine Learning (ML) research coming largely from increasing compute budgets, Richard Sutton coined the idea of a "bitter lesson", wherein more computational power will ultimately supersede a hand-crafted solution (Sutton, 2019). More recently, increasing compute power on a general purpose architecture has also shown to be wildly successful in the Natural Language Processing (NLP) community (Vaswani et al., 2017; Wei et al., 2022). In particular, emergent capabilities in very large language models (vLLMs) have made it possible to approach a variety of tasks wherein only few (if any) samples are labeled, and no further fine-tuning on task-specific data is required at all.

In stark contrast to the complex pipelines in modern lexical simplification systems (Ferrés et al., 2017; Qiang et al., 2020; Štajner et al., 2022), we present a simplistic approach utilizing few-shot prompts based on a vLLM with basic instructions on simplification, which returns frustratingly good results considering the overall complexity of the approach, which utilizes a grand total of four hand-labeled instances. We present our results on the TSAR-2022 shared task (Saggion et al., 2022), which evaluates lexical simplification systems in three available languages (English, Spanish and Portuguese), with ten labeled instances and around 350 unlabeled test samples provided per language. For the English subset, official results rank our model as the best-performing submission, indicating that this approach may be another instance of the bitter lesson. While the initial findings are indeed promising, we want to carefully evaluate erroneous instances on the test set to analyze potential pitfalls, and further detail some of our experiences in hand-crafting prompts. We also acknowledge the technical challenges in reproducing (and deploying) systems based on vLLMs, especially given that suitable models exceed traditional computing budgets.

2 Prompt-based Lexical Simplification

With the public release of the GPT-3 language model (Brown et al., 2020), OpenAI has started the run on a series of now-available vLLMs for general-purpose text generation (Thoppilan et al., 2022; BigScience, 2022; Zhang et al., 2022). Across these models, a general trend in scaling beyond a particular parameter size can be observed, while keeping the underlying architectural design close to existing smaller models. Through exhibiting zero-shot transfer capabilities, such models have also become more attractive for lower-resourced tasks; oftentimes, models are able to answer questions formulated in natural language with somewhat sen-

[1] https://github.com/dennlinger/TSAR-2022-Shared-Task

sible results. Particular template patterns (so-called *prompts*) are frequently used to guide models towards predicting a particularly desirable output or answer format, without requiring a dedicated training on labeled examples.

Utilizing this paradigm shift, we experimented with different prompts issued to OpenAI's largest available model, `text-davinici-002`, which totals 176B parameters. Our first approach uses a singular prompt template in a zero-shot setting to obtain predictions for the shared task; we further improve upon these results by combining predictions from different prompt templates later on.

2.1 Run 1: Zero-shot Prediction

Upon inspecting the provided trial data, we noted that the simplification operations required a vastly different contextualization within the provided sample sentence. Whereas some instances can be solved with pure synonym look-ups (e.g., "compulsory" and "mandatory"), others require a more nuanced look at the context sentence (e.g., replacing "disguised" with "dressed"). To avoid biasing system predictions by providing samples as a prompt template, we provide a baseline that is entirely based on a single zero-shot query; it provides the context sentence and identifies the complex word, asking the model for ten simplified synonyms of the complex word in the given context. Given that no additional knowledge is provided to the model, the zero-shot contextual query also provides a reasonable lower-bound for the task setting. A secondary advantage of minimal provided context in zero-shot settings is the reduced computational cost, which will be discussed in more detail in Section 3.4.

2.2 Filtering Predictions

Model suggestions are returned as free-form text predictions, generally in the form of comma-separated lists or enumerations. This requires the additional step of parsing the output prediction into the more structured ranked predictions required for the shared task, which varies between the models used. In our experience, no clear pattern can be expected from the model and seems to be non-deterministic even with set template structures. We additionally employ a list of simple filters to ensure the quality of predictions, as detailed in Appendix C. The resulting model suggestions are considered in ranked order, and no prediction confidence scores or similar information was used to re-rank single-prompt predictions.

2.3 Run 2: Ensemble Predictions

Upon inspecting the results from the first run, we noticed that in some instances, predictions were almost fully discarded due to filtering. Simultaneously, we had already previously encountered strong variability in system generations when changing the prompt template or altering the context setting. To this extent, an ensemble of predictions from multiple different prompt templates was utilized to broaden the spectrum of possible generations, as well as ensuring that a minimum number of suggestions survives the filtering step.

2.3.1 Prompt Variations

The exact prompts are detailed in Table 3. Utilized variations can be grouped into *with context* (the context sentence is provided), or *without context* (synonyms are generated from the complex word alone). Simultaneously, different prompts also contain between zero and two examples taken from the trial data, including their expected outputs. This can be interpreted as a few-shot setting in which the model is demonstrated on what correct answers may look like for the particular task. We further vary the generation temperature, where a higher value increases the likelihood of a more creative (but not always correct) prediction, enabling a more diverse candidate set.

2.3.2 Combining Predictions

For each of the six prompts p, we ask the model to suggest ten alternative simplified expressions S_p and filter them with the exact same rules as the single prompt system in Run 1. In order to combine and re-rank suggestions s, we assign a combination score V to each distinct prediction $s \in \bigcup_p S_p$:

$$V(s) = \sum_p \max(5.5 - 0.5 \cdot \text{rank}_{S_p}(s), 0), \quad (1)$$

where $\text{rank}_{S_p}(s)$ is the ranked position of suggestion s in the resulting ranking from prompt p. If $s \notin S_p$, we set $\text{rank}_{S_p}(s) = \infty$. The scaling parameters are chosen arbitrarily and can be adjusted to account for the expected number of suggestions per prompt. We estimate that the biggest performance improvement is coming simply from providing enough predictions post filtering. As a secondary gain, we see more consistent behavior in the top-most prediction slots, boosting especially the @1 performance of the ensemble.

Run	ACC@1	Acc@k@Top1			MAP@k			Potential@k		
		$k=1$	$k=2$	$k=3$	$k=3$	$k=5$	$k=10$	$k=3$	$k=5$	$k=10$
Ensemble (Ours)	**0.8096**	**0.4289**	**0.6112**	**0.6863**	**0.5834**	**0.4491**	**0.2812**	**0.9624**	**0.9812**	**0.9946**
Single (Ours)	0.7721	0.4262	0.5335	0.5710	0.5090	0.3653	0.2092	0.8900	0.9302	0.9436
MANTIS-1	0.6568	0.319	0.4504	0.5388	0.473	0.3599	0.2193	0.8766	0.9463	0.9785
UoM&MMU-1	0.6353	0.2895	0.4530	0.5308	0.4244	0.3173	0.1951	0.8739	0.9115	0.9490
LSBert	0.5978	0.3029	0.4450	0.5308	0.4079	0.2957	0.1755	0.8230	0.8766	0.9463
TUNER	0.3404	0.1420	0.1689	0.1823	0.1706	0.1087	0.0546	0.4343	0.4450	0.4450

Table 1: Results on the English language test set of the TSAR-2022 shared task, ranked by *ACC@1* scores. Listed are our own results (*Ensemble* and *Single*), the two best-performing competing systems (*MANTIS* and *UoM&MMU*), as well as provided baselines (*LSBert* (Qiang et al., 2020) and TUNER (Ferrés et al., 2017)).

3 Results and Limitations

3.1 Results for English

For the official runs, we initially only submitted predictions for the English subset; an excerpt of the results can be seen in Table 1. While the zero-shot single prompt run has consistently better results on most metrics, it does not outperform all systems for large candidate sets; e.g., Potential@10 is lower than that of competing approaches, including the LSBert baseline. We attribute this to the previously mentioned issue of filtering predictions, and can see a consequent improvement especially for larger k by using the proposed ensemble method. Here, the Potential@10 scores indicate that at least one viable prediction is present in *all but three samples*.

3.2 Results for Spanish and Portuguese

Given the surprisingly good results on the English subset, we decided to extend our experiments to the Spanish and Portuguese tracks as well. Transferring the prompts to Spanish or Portuguese is surprisingly simple. We alter the prompt to: *"Given the above context, list ten alternative* ***Spanish*** *words for* `'complex_word'` *that are easier to understand."* (bold highlight indicates change).
Without this adaption, returned suggestions generally tend to be in English, which could be an attractive opportunity to mine cross-lingual simplifications in future work. By adding the output language explicitly, we ensure that the suggestions match the expected results. For Portuguese, the prompt can be adapted accordingly.
We find that our system also outperforms all competing submitted approaches in the shared task; result comparisons can be found in Table 4 and 5 in the Appendix, respectively. Notably, predictions for Portuguese perform slightly better, which goes against intuition, given that Spanish is usually a highly represented language in multilingual corpora. We suspect that a more literal wording of synonyms in Portuguese, compared to multi-word expressions in Spanish, could be the cause.

3.3 Error Analysis

As is common for sequence-to-sequence tasks, crafting an approach centered around a LM requires consideration of the particular challenges arising. We detail some of the errors we have encountered in our predictions that are unlikely to appear in more stringently designed pipelines. Instances for particular failure cases can be found in Table 2.

Unstable Prompts One of the primary challenges, particularly for zero-shot prompt settings, is the unreasonable variance observed in results based on even just slightly altered prompt templates. For example, when removing the explicit mention of *Context:*, *Question:* and *Answer:* in the prompt template, the model is frequently predicting fewer than the ten requested answers. Practical limitations in our computational budget also mean that we have no guarantee that these prompts are yielding the best possible results; given the variability, multiple runs should be compared for a thorough pattern of a "best" prompt.

Lack of Context Instances with longer (or subtly enforced) context cues show issues where these hints are not properly recognized. In Table 2, we can see the model changing the term "*collision*" to a particular mode of transportation, such as "*car crash*", while an explicit context clue is given through the word "*flight*" in the original sentence.

Enforcing Language While the transfer to Spanish and Portuguese is largely successful, the model's capabilities seem to be still limited in maintaining the language throughout all samples.

Error Type	Context (complex word in bold)	Model Predictions
Lack of Context	#7-8 Despite the fog, other flights are reported to have landed safely leading up to the **collision**.	car crash, train wreck, ...
Hallucinations	The larva grows to about 120-130 mm, and **pupates** in an underground chamber.	Transforms into a pupa, ...
Language	[...] propiciado la **decadencia** de la Revolución francesa.	decline, deterioration, ...

Table 2: Instances of observed failure classes in our system's predictions.

For instances with particularly rare complex terms, the predictions are sometimes still in English, despite the specific prompt request to return Spanish/Portuguese results.

Hallucinations The necessity for post-filtering of suggestions stems largely from the spontaneous occurrence of hallucinations in responses. While hallucinations in vLLMs are less about invalid vocabulary terms, we observe instances where unnecessary multi-word suggestions were returned, or random inflections (such as the infinitive form with an additional "to") were generated.
Similar to the issues with language enforcing, this occurs more frequently with particularly complex words; in this sense, the system conversely fails at instances that are most in need of simplification. However, we note that some of the generated multi-word expressions are actually more helpful for understanding, even though the generations are not precisely matching expected outputs.

3.4 Computational Limitations

Running a vLLM in practice, even for inference-only settings, is non-trivial and requires compute resources that are far beyond many public institution's hardware budget. For the largest models with publicly available checkpoints[2], a total of around **325GB GPU memory** is required, assuming efficient storage in `bfloat16` or similar precision levels. The common alternative is to obtain predictions through a (generally paid) API, as was the case in this work. Especially for the ensemble model, which issues six individual requests to the API per sample, this can further bloat the net cost of a single prediction. To give context of the total cost, we incurred a total charge of slightly over $7 for computing predictions across the entire test set of 373 English samples, which comes out to about 1000 tokens per sample, or around $0.02 at the current OpenAI pricing scheme.[3] For the Spanish subset and language-dependent prompt development, the total cost came to about $10, primarily due to longer sample contexts. Costs for Portuguese processing were around $6.50. While the singular prompt approach is cheaper at around 1/6 of the total cost, even then a continuously deployed model has to be supplied with a large enough budget. Aside from monetary concerns, environmental impacts are also to be considered for larger-scale deployments of this kind (Lacoste et al., 2019).

4 Conclusion and Future Work

Utilizing prompted responses from vLLMs seems to be a promising direction for lexical simplification; particularly in the constrained setting with pre-identified complex words the model performs exceptionally well, even when presented with a severely restricted budget of labeled training data. While the approach also offers promising directions for multi- and cross-lingual approaches, obtaining state-of-the-art results in other languages, we are presented with a prohibitive amount of computation per sample instance. It would therefore be an interesting addition to deal with resource-constraint systems, putting the prediction power into a slightly different perspective. Finally, we are also reminded of the unpredictable nature of neural LMs; given similar inputs, prediction quality can vary greatly between samples, including a complete breakdown in performance.
For future work, we are considering approaches to generate static resources from vLLMs (Schick and Schütze, 2021), which may require only a one-time commitment to spending on datasets, which can then used as training data for cheaper systems. Exploration of prompt tuning approaches for automated search of suitable prompt templates would also greatly accelerate the development process of domain-specific applications (Lester et al., 2021).

[2] At the time of writing, this would be the 176B Bloom model (BigScience, 2022), which has a similar parameter count to OpenAI's davinci-002 model.

[3] `https://openai.com/api/pricing/`, last accessed: 2022-10-01

References

Workshop BigScience. 2022. BLOOM (revision 4ab0472).

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Daniel Ferrés, Horacio Saggion, and Xavier Gómez Guinovart. 2017. An adaptable lexical simplification architecture for major Ibero-Romance languages. In *Proceedings of the First Workshop on Building Linguistically Generalizable NLP Systems*, pages 40–47, Copenhagen, Denmark. Association for Computational Linguistics.

Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. 2019. Quantifying the carbon emissions of machine learning. *CoRR*, abs/1910.09700.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jipeng Qiang, Yun Li, Zhu Yi, Yunhao Yuan, and Xindong Wu. 2020. Lexical simplification with pretrained encoders. *Thirty-Fourth AAAI Conference on Artificial Intelligence*, page 8649–8656.

Horacio Saggion, Sanja Štajner, Daniel Ferrés, Kim Cheng Sheang, Matthew Shardlow, Kai North, and Marcos Zampieri. 2022. Findings of the tsar-2022 shared task on multilingual lexical simplification. In *Proceedings of TSAR workshop held in conjunction with EMNLP 2022*.

Timo Schick and Hinrich Schütze. 2021. Generating datasets with pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6943–6951, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Richard Sutton. 2019. The bitter lesson. *Incomplete Ideas (blog)*, 13:12.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, YaGuang Li, Hongrae Lee, Huaixiu Steven Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Yanqi Zhou, Chung-Ching Chang, Igor Krivokon, Will Rusch, Marc Pickett, Kathleen S. Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Soraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Diaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravi Rajakumar, Alena Butryna, Matthew Lamm, Viktoriya Kuzmina, Joe Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Aguera-Arcas, Claire Cui, Marian Croak, Ed H. Chi, and Quoc Le. 2022. Lamda: Language models for dialog applications. *CoRR*, abs/2201.08239.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Sanja Štajner, Daniel Ferrés, Matthew Shardlow, Kai North, Marcos Zampieri, and Horacio Saggion. 2022. Lexical simplification benchmarks for English, Portuguese, and Spanish. *Frontiers in Artificial Intelligence*, 5.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *CoRR*, abs/2206.07682.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: open pre-trained transformer language models. *CoRR*, abs/2205.01068.

A Prompt Templates

Table 3 provides the exact prompt templates used in the submission. Notably, the *zero-shot with context* prompt is included twice, but with different generation temperatures; with this we increase the likelihood of strong candidates being retained. For few-shot prompts, we have taken samples from the previously published trial set for the respective language. In instances where less than 10 distinctly different suggestions were provided by annotators, we manually extended the list of examples to match exactly ten results based on our own judgment. For instances with more provided suggestions, we limit ourselves to the ten most frequently occurring ones. The reason for this is that GPT-3 otherwise tended to return an inconsistent number of suggestions in our preliminary testing. The exact prompts for the Spanish and Portuguese runs can be found in our repository.

B Hyperparameters

We use the OpenAI Python package[4] version 0.23.0 for our experiments. For generation, the function `openai.Completion.create()` is used, where most hyperparameters remain fixed across all prompts. We explicitly list those hyperparameters below that differ from their respective default values.

1. `model="text-davinci-002"`, which is the latest and biggest available model for text completion.

2. `max_tokens=256`, to ensure sufficient room for generated outputs. In practice, most completions are vastly below the limit.

3. `frequency_penalty=0.5`, as well as `presence_penalty=0.3`, which jointly penalize present tokens and token repetitions. The values are well below the maximum (values can range from -2 to 2), since individual subword tokens might indeed be present several times across multiple (valid) predictions. A more detailed computation can be found in the documentation of OpenAI.[5]

Outside of the repetition penalties, the most influential parameter choice for generation is the sampling temperature. We generally take a more measured approach than the default (`temperature=1.0`), but vary temperature across our ensemble prompts to ensure a more diverse result set overall. We list the used temperatures in Table 3. *Zero-shot with context* is used twice in the ensemble, once with a more conservative temperature, and once with a more "creative" (higher) temperature. For the singular prompt run, we use the conservative *zero-shot with context* variant.

C Post-Filtering Operations

Given the uncertain nature of predictions by a language model, we employ a series of post-filtering steps to ensure high quality outputs. This includes stripping newlines/spaces/punctuation (`\n :;.?!`), lower-casing, removing infinitive forms (in some instances, we observed predictions in the form of "to deploy" instead of simply "deploy"), as well as removing identity predictions (e.g., the prediction being the same as the original complex word) and deduplicating suggestions. Additionally, we noticed that for some instances, generated synonyms resemble more of a "description" rather than truly synonymous expressions (example: "people that are crazy" as a suggestion for "maniacs"). Given the nature of provided data, we removed extreme multi-word expressions (for English, any suggestion with more than two words, for Spanish and Portuguese more than three words in a single expression).

[4] https://github.com/openai/openai-python

[5] https://beta.openai.com/docs/api-reference/parameter-details

Prompt Type	Template

Zero-shot with context
temperature:
0.3 (conservative),
0.8 (creative)
	`Context: {context_sentence}\n`
	`Question: Given the above context, list ten alternatives for “{complex_word}” that are easier to understand.\n`
	`Answer:`

Single-shot with context
temperature: 0.5
	`Context: A local witness said a separate group of attackers disguised in burqas – the head-to-toe robes worn by conservative Afghan women – then tried to storm the compound.\n`
	`Question: Given the above context, list ten alternative words for “disguised” that are easier to understand.\n`
	`Answer:\n1. concealed\n2. dressed\n3. hidden\n4. camouflaged\n5. changed\n6. covered\n7. masked\n8. unrecognizable\n9. converted\n10. impersonated\n\n`
	`Context: {context_sentence}\n`
	`Question: Given the above context, list ten alternatives for “{complex_word}” that are easier to understand.\n`
	`Answer:`

Two-shot with context
temperature: 0.5
	`Context: That prompted the military to deploy its largest warship, the BRP Gregorio del Pilar, which was recently acquired from the United States.\n`
	`Question: Given the above context, list ten alternative words for “deploy” that are easier to understand.\n`
	`Answer:\n1. send\n2. post\n3. use\n4. position\n5. send out\n6. employ\n7. extend\n8. launch\n9. let loose\n10. organize\n\n`
	`Context: The daily death toll in Syria has declined as the number of observers has risen, but few experts expect the U.N. plan to succeed in its entirety.\n`
	`Question: Given the above context, list ten alternative words for “observers” that are easier to understand.\n`
	`Answer:\n1. watchers\n2. spectators\n3. audience\n4. viewers\n5. witnesses\n6. patrons\n7. followers\n8. detectives\n9. reporters\n10. onlookers\n\n`
	`Context: {context_sentence}\n`
	`Question: Given the above context, list ten alternatives for “{complex_word}” that are easier to understand.\n`
	`Answer:`

Zero-shot w/o context temperature: 0.7	`Give me ten simplified synonyms for the following word: {complex_word}`

Single-shot w/o context
temperature: 0.6
	`Question: Find ten easier words for “compulsory”.\n`
	`Answer:\n1. mandatory\n2. required\n3. essential\n4. forced\n5. important\n6. necessary\n7. obligatory\n8. unavoidable\n9. binding\n10. prescribed\n\n`
	`Question: Find ten easier words for “{complex_word}”.\n`
	`Answer:`

Table 3: The English prompt templates used for querying the OpenAI model, including associated generation temperatures. Only written out “\n” symbols indicate newlines, visible line breaks are inserted for better legibility. Only top-most prompt template with conservative temperature was used in the single prompt (Run 1), as well as in the ensemble run (Run 2). All other prompts‘ were only included in the ensemble submission.

Run	ACC@1	Acc@k@Top1			MAP@k			Potential@k		
		$k=1$	$k=2$	$k=3$	$k=3$	$k=5$	$k=10$	$k=3$	$k=5$	$k=10$
Ensemble (Ours)	**0.6521**	**0.3505**	**0.5108**	**0.5788**	**0.4281**	**0.3239**	**0.1967**	**0.8206**	**0.8885**	**0.9402**
Single (Ours)	0.5706	0.3070	0.3967	0.4510	0.3526	0.2449	0.1376	0.6902	0.7146	0.7445
PresiUniv-1	0.3695	0.2038	0.2771	0.3288	0.2145	0.1499	0.0832	0.5842	0.6467	0.7255
UoM&MMU-3	0.3668	0.1603	0.2282	0.269	0.2128	0.1506	0.0899	0.5326	0.6005	0.6929
LSBert	0.2880	0.0951	0.1440	0.1820	0.1868	0.1346	0.0795	0.4945	0.6114	0.7472
TUNER	0.1195	0.0625	0.0788	0.0842	0.0575	0.0356	0.0184	0.144	0.1467	0.1494

Table 4: Results on the Spanish language test set of the TSAR-2022 shared task, ranked by *ACC@1* scores. Listed are our own results (*Ensemble* and *Single*), the two best-performing competing systems (*PresiUniv* and *UoM&MMU*), as well as provided baselines (*LSBert* (Qiang et al., 2020) and TUNER (Ferrés et al., 2017)).

Run	ACC@1	Acc@k@Top1			MAP@k			Potential@k		
		$k=1$	$k=2$	$k=3$	$k=3$	$k=5$	$k=10$	$k=3$	$k=5$	$k=10$
Ensemble (Ours)	**0.7700**	**0.4358**	**0.5347**	**0.6229**	**0.5014**	**0.3620**	**0.2167**	**0.9171**	**0.9491**	**0.9786**
Single (Ours)	0.6363	0.3716	0.4625	0.5160	0.4105	0.2889	0.1615	0.7860	0.8181	0.8422
GMU-WLV-1	0.4812	0.2540	0.3716	0.3957	0.2816	0.1966	0.1153	0.6871	0.7566	0.8395
Cental-1	0.3689	0.1737	0.2433	0.2673	0.1983	0.1344	0.0766	0.524	0.5641	0.6096
LSBert	0.3262	0.1577	0.2326	0.286	0.1904	0.1313	0.0775	0.4946	0.5802	0.6737
TUNER	0.2219	0.1336	0.1604	0.1604	0.1005	0.0623	0.0311	0.2673	0.2673	0.2673

Table 5: Results on the Portuguese language test set of the TSAR-2022 shared task, ranked by *ACC@1* scores. Listed are our own results (*Ensemble* and *Single*), the two best-performing competing systems (*GMU-WLV* and *Cental*), as well as provided baselines (*LSBert* (Qiang et al., 2020) and TUNER (Ferrés et al., 2017)).