The Bitter Lesson Strikes Again: 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. In this technical report, 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 2022 TSAR shared task consists of an "ensemble" of six different prompt templates with varying context levels. Aside from detailing the implementation and setup, we spend the remainder of this work discussing the particularities of generated suggestions and implications for future work.

1 Introduction

Across the general landscape of AI research, Richard Sutton coined the idea of a "bitter lesson", wherein more computational power will ultimately supersede a hand-crafted solution TODO: cite bitter lesson. While we have previously seen this primarily for Computer Vision research TODO: cite ImageNet and GoogleNet or something, more recently, increasing compute has also shown to be wildly successful in the NLP community TODO: cite emergent behavior paper. In particular, emergent capabilities in extremely large language models (xLLMs) have made it possible to approach a variety of tasks wherein only few (if any) samples are provided with labels, and no further fine-tuning is required at all.

In stark contrast to the complex setup required for modern lexical simplification systems TODO: cite context work, in this work we present a simplistic pipeline based only on prompting a xLLM for lexical simplification, which returns frustratingly good results. After submitting to the TSAR 2022 shared task for English, standings indicate that this approach may be another instance of the bitter lesson, producing results strictly better than alternative solutions across all evaluated metrics.

While the initial findings are indeed promising, we want to cautiously evaluate erroneous instances on the test set to analyze potential pitfalls, and further detail some of our experiences in crafting prompts for context-dependent prediction tasks. Furthermore, we acknowledge the technical challenges in reproducing (and productionizing) systems based on xLLMs, especially given that suitable models exceed traditional computing budgets.

TODO: somewhere I need to actually mention more details on the TSAR shared task and task data!! This could be enough content for a Background/Rel Work section?

2 Prompt-based Lexical Simplification

TODO: Write about what the general idea is for this model. Touted as the TODO: insert flashy phrase, such as "human-level AI", the GPT-3 model released by OpenAI TODO: cite has been the first in a series of available xLLMs for general-purpose text generation TODO: cite other works, like BLOOM, OPT, LamBda. 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 TODO: maybe cite something. After training on enough data for long enough periods, these models exhibit zero-shot transfer capabilities across a wide range of tasks; more precisely, models are able to answer questions formulated in natural language with (more or less) sensible results. Particular template patterns (so-called *prompts*) are frequently used to guide models towards predicting a particularly desirable output or answer format.

Using this paradigm, we experimented with different prompts issued to OpenAI's largest available model, `text-davinici-002`, which totals 176B

parameters and is available through the OpenAI API[1]. For the final prompts used in our submission, please refer to Appendix A. We also detail any further hyperparameters and filtering steps used in the pipeline. Furthermore, we submitted two different runs for the TSAR 2022 shared task, for which we will detail the differences below.

2.1 Run 1: Zero-shot Prediction

TODO: Here, we explain the basic setup of the prompting approach, and detail basic hyperparameters (ten responses per prompt that we ask the system to report). TODO: Also explain why we use the particular zero-shot prompting approach. Primarily also say that we are limited in the compute budget and the evaluation strategies on trial data with few-shot approaches. We estimate that this serves as a reasonable "lower-bound" submission. This is especially the case if we only consider pure zero-context approaches for some of the models

2.2 Filtering Predictions

TODO: Write about how we need to perform basic post-filtering because the output is not always the same. Give maybe an example in a figure? The full list of filtering operations is detailed in Appendix C.

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 variance in system generations when varying 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 filtering in each instance.

2.3.1 Prompt Variants

The added prompts are further detailed in Table 2, but can be generally grouped into *with context* (the context sentence is provided), or *without context* (only the complex word is given and synonyms should be generated from there). Simultaneously, different prompts also contain between zero and two examples with expected outputs for instances from the trial data, to demonstrate the system what correct answers may look like. Aside from this, we vary the generation temperature, where a higher temperature increases the likelihood of a potentially more creative (but not always correct) prediction.

2.3.2 Combining Predictions

For each of the six prompts $p \in P$, we ask the model to generate ten individual suggestions and filter them with the exact same rules as the system in Run 1. In order to combine and re-rank suggestions S, we assign the following score to each distinct prediction $s \in S$: TODO: rewrite this shit to make more sense

$$\text{score}(s) = \sum_{p \in P} 5.5 - 0.5 \cdot \text{rank}_p(\text{s}), \qquad (1)$$

TODO: Extending this, we can obtain some of the predictions by combining the ranks across the different ensemble models

3 Error Analysis and Limitations

As with other sequence-to-sequence tasks, the output of a xLLM cannot be guaranteed to be entirely correct at all times. In this section, we detail some of the particular challenges we have encountered during the design process.

3.1 Computational Budgeting

Running a xLLM in practice, even for inference-only settings, is non-trivial and requires compute that is far beyond many public institution's hardware budget. For the largest models with publicly available checkpoints[2], a total of around 320GB **GPU memory** is required.

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 $7.15 for computing predictions across the entire test set of 373 English samples for the shared task, which comes to about 1000 tokens per sample, or around $0.02 at the current OpenAI pricing scheme.[3]

4 Conclusion and Future Work

TODO: Maybe ask Matthew/organizers if we can actually get the predictions for the baselines? This

[1] TODO: URL link

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

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

Command	Output	Command	Output
{\"a}	ä	{\c c}	ç
{\^e}	ê	{\u g}	ğ
{\`i}	ì	{\l}	ł
{\.I}	İ	{\~n}	ñ
{\o}	ø	{\H o}	ő
{\'u}	ú	{\v r}	ř
{\aa}	å	{\ss}	ß

Table 1: Results on the English language test set of the TSAR shared task. Listed are our own results (*UniHD*), the two best-performing competing systems TODO: include, as well as provided baselines TODO: name them and cite the relevant paper.

way, we could compare the approaches. The context paper also mentions a few instances, so maybe look there?

References

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L_1-regularized log-linear models. In *Proceedings of the 24th International Conference on Machine Learning*, pages 33–40.

James W. Cooley and John W. Tukey. 1965. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.

Mohammad Sadegh Rasooli and Joel R. Tetreault. 2015. Yara parser: A fast and accurate dependency parser. *Computing Research Repository*, arXiv:1503.06733. Version 2.

A Prompt Templates

Table 2 provides the exact prompt templates used in the submission. Notably, the *zero-shot with context* prompt is included twice, but with different generation temperatures TODO: mention which ones, or link to the hyperparameter table?.

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 the default values. TODO: Pretty sure that most of these also constitute the default parameters? Only include those that activelsy differ!!

1. `model="text-davinci-002"`, the latest and biggest model for text generation at the time of writing,
2. `max_tokens=256`, to limit generation length,
3. `top_p=1.0`, to include the entire vocabulary during token generation,
4. `best_of=1`, TODO: I think default?,
5. `frequency_penalty=0.5`, TODO: read up on this parameter again? How is it different from the other one?,
6. `presence_penalty=0.3`, which penalizes already present tokens. We choose a lower value, since individual subword tokens might indeed be present several times across multiple (valid) predictions TODO: maybe explain with an example?.

TODO: Include table with the associated temperatures.

C Post-Filtering Operations

TODO: Include the list of operations by which we filter.

[4] TODO: Insert package URL

Prompt Type	Template

Zero-shot with context	`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	`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	`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	`Give me ten simplified synonyms for the following word: {complex_word}`

Single-shot w/o context	`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 2: The exact prompt templates used for querying the OpenAI model. Only written out \n indicate newlines, visible newlines are inserted for better legibility. The top-most prompt template was used in both runs. The remaining prompts were only included in the ensemble submission (Run 2).